

Artificial Intelligence for Cybersecurity

Week 6: Threat Analysis & Adversarial AI

James Hu

Ph.D. Candidate

Acknowledgements: Dr. Hsinchun Chen



Online M.S. in
AI for Business

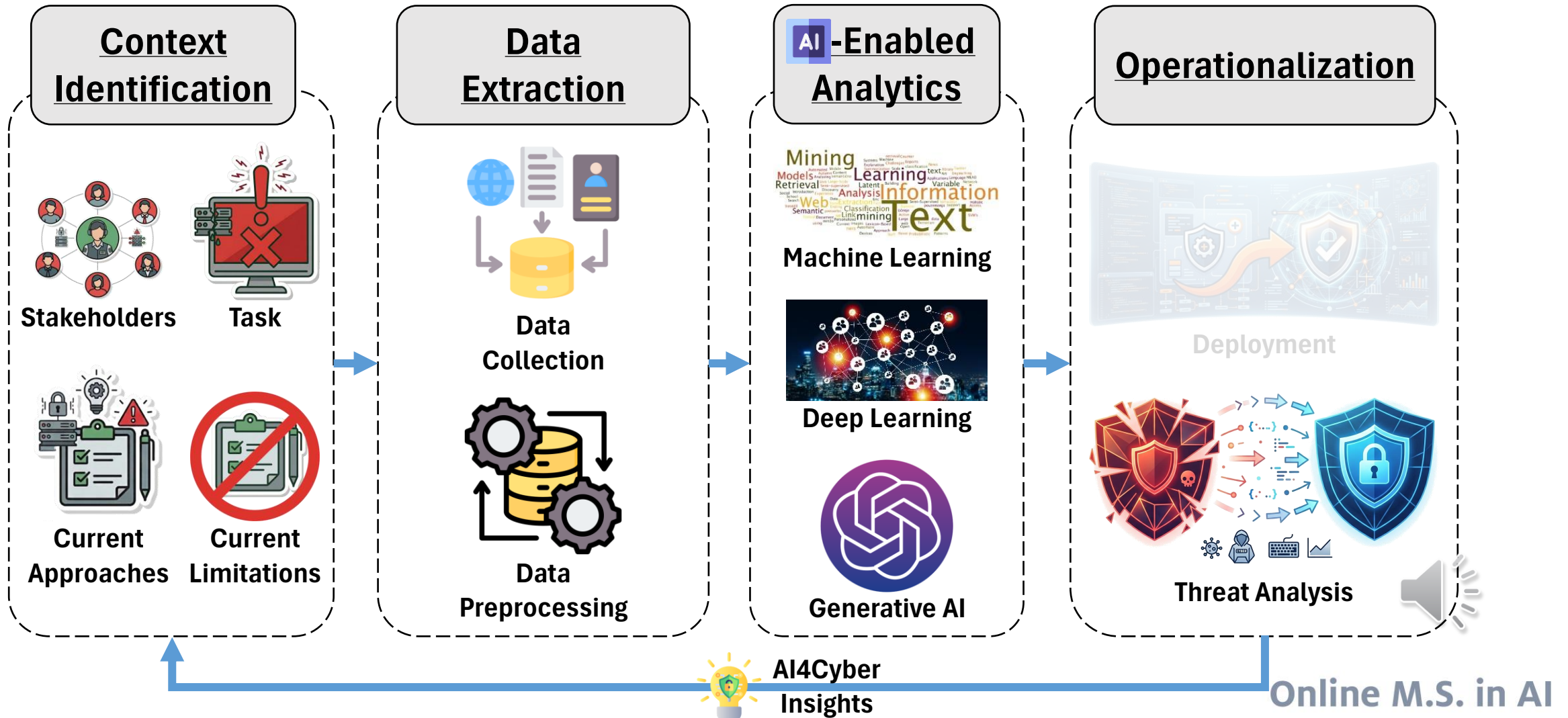


Today's Learning Objectives

- ▶ Explain the difference between model finetuning and model alignment.
- ▶ Identify the different types of misalignment prevalent in cybersecurity
- ▶ Understand the three core steps of adversarial AI for threat analysis.

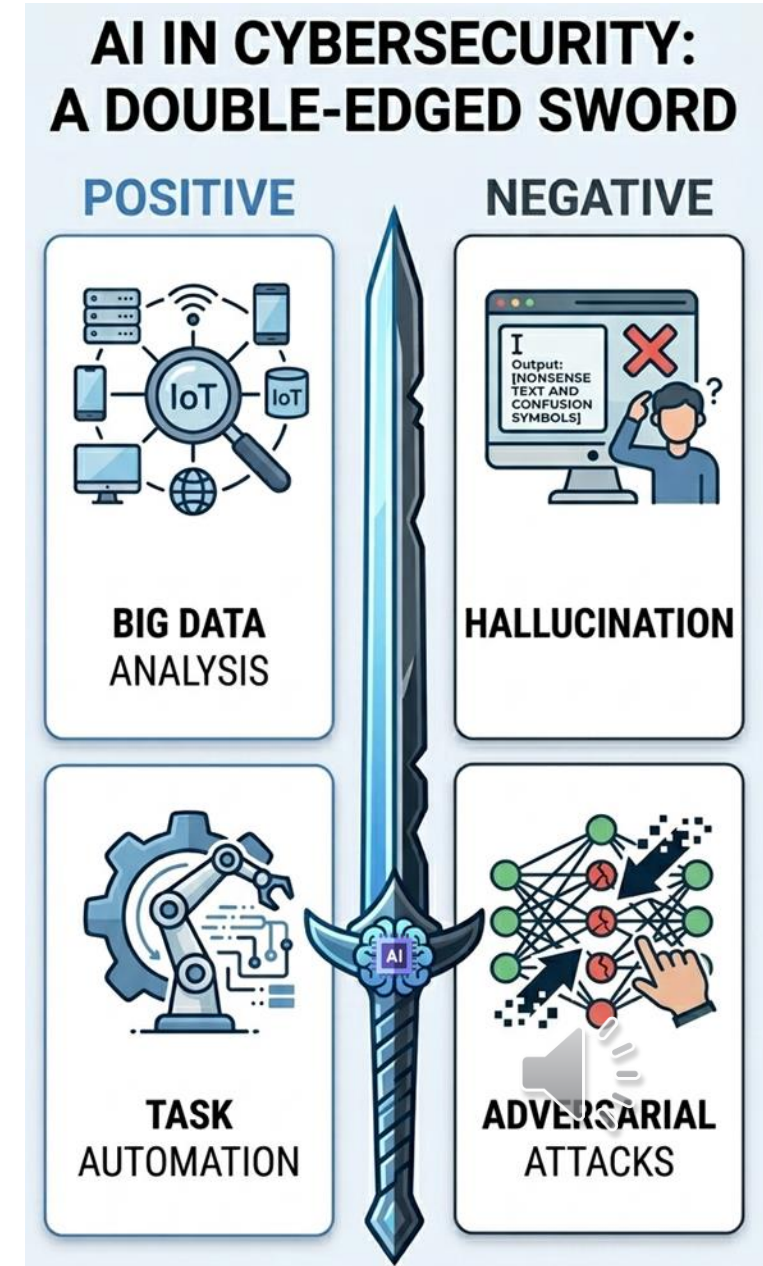


From Week 1: AI4Cyber Lifecycle



From Week 1: The Double-Edged Sword

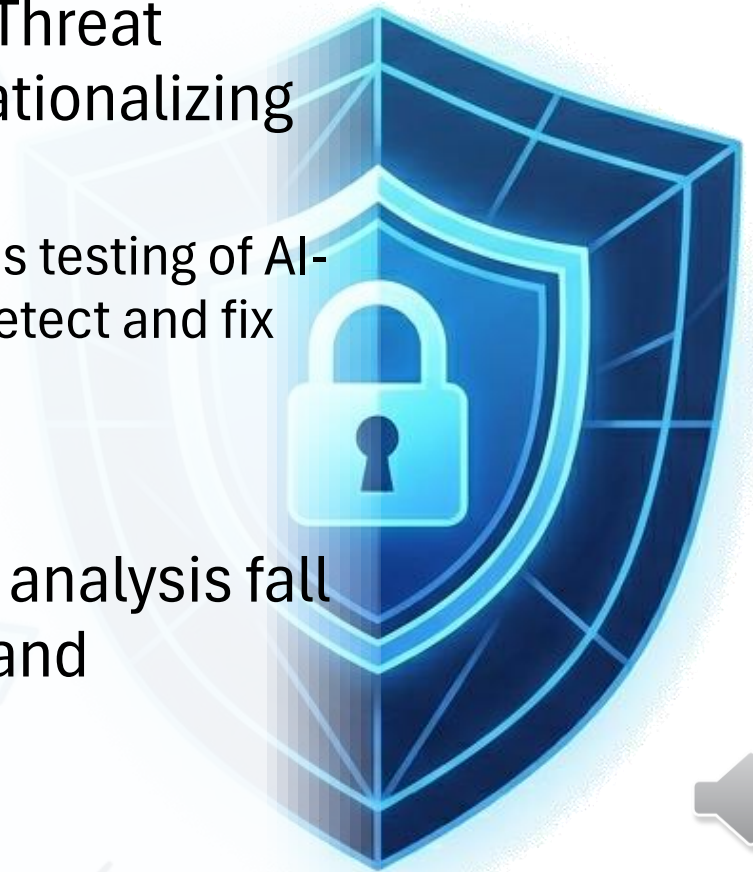
- ▶ AI itself is also a digital technology.
 - ▶ How “secure” AI systems are must also be considered.
- ▶ AI models often make mistakes by itself due to insufficient or incorrect training.
 - ▶ Difficult to detect during training and often require specific evaluation after developing the AI-enabled analytic (Operationalization Component).
- ▶ Cybersecurity adds another layer.
 - ▶ Unlike other fields, cybersecurity include the adversary, who aims to purposely induce mistakes in cybersecurity systems.



Operationalization: Threat Analysis

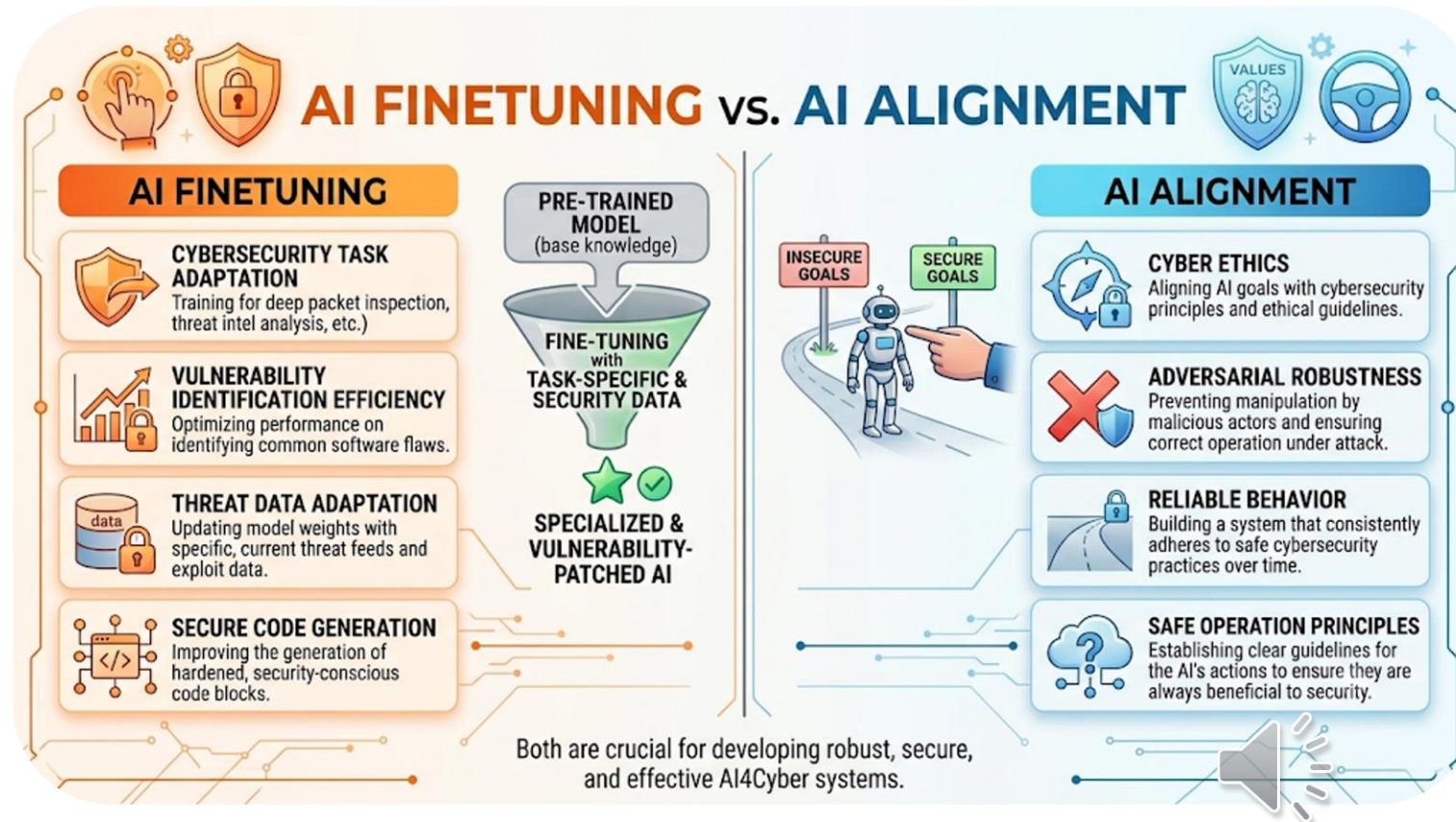


- ▶ The AI4Cyber lifecycle includes Threat Analysis as another way of operationalizing AI-enabled analytics.
 - ▶ **Threat Analysis:** Specialized stress testing of AI-enabled analytics to proactively detect and fix flaws.
- ▶ Evaluation techniques for threat analysis fall under the fields of AI alignment and Adversarial AI.
 - ▶ Can be manual or automated



AI Alignment

- ▶ AI Alignment is a field of research aiming to “align” AI systems with human values.
 - ▶ Different from AI finetuning.
- ▶ Determining how aligned an AI system is during development is difficult.
 - ▶ Evaluating and adjusting how the system behaves after development is easier.



AI Alignment

- ▶ AI alignment contains several types of alignment, each aligning AI systems with a different human value.
 - ▶ Each contain respective misalignments, or the behaviors we don't want from AI systems.

National Institution for Standards and Technology's AI Risk Management Framework 1.0 (NIST's AI-RMF 1.0)

Aligned Value	Description	Example Misalignment
Validity	AI outputs are accurate and correct	A decision tree-based malware detector outputs many false positives by misclassifying benign files.
Safety	AI outputs do not cause harm or endangering humans	A code remediation agent generates code for malware files.
Privacy	AI systems do not unintentionally disclose private data	An AI phishing detector outputs private emails it saw during its training.
Explainability	How AI systems reach its decisions are easily understandable to users	An AI-based vulnerability scanner flags a critical patch as malware without explanation
Fairness	AI systems do not unnecessarily discriminate against a group.	An AI agent monitoring network traffic disproportionately prevents users from specific geographic regions

**Applicable to
ML, DL, and AAI
systems**

**Applicable to
DL and AAI
systems**

**Focus of
this class**

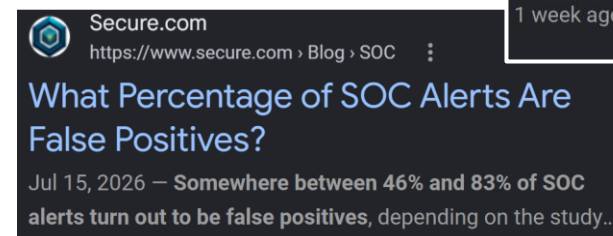
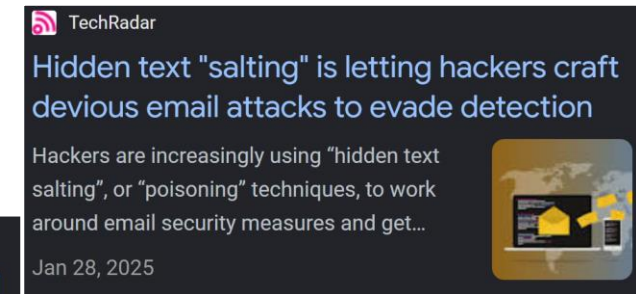
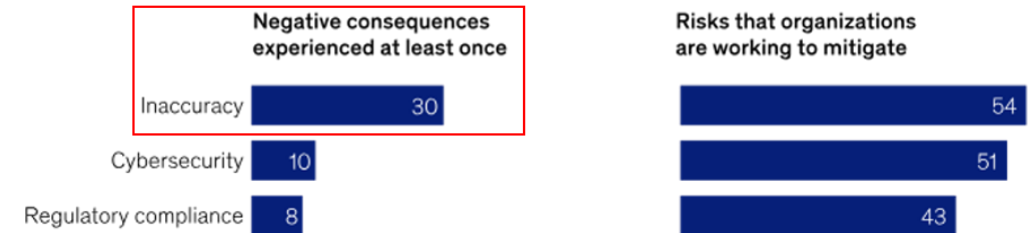


M.S. in AI

AI Alignment: Validity Alignment

- ▶ Validity alignment assures that an AI system functions **accurately** and precisely as intended for its specific use case.
 - ▶ Most fundamental type of AI alignment.
- ▶ Applicable to **all** types of AI-enabled analytics (ML, DL, and AAI).
 - ▶ In turn, validity misalignment is the most prevalent of all misalignment types.
- ▶ Validity misaligned AI4Cyber systems can severely compromise cybersecurity.

Inaccuracy is the AI-related risk that respondents most often say their organizations have experienced and are working to mitigate.



Online M.S. in AI

AI Alignment: Validity Alignment

- ▶ Validity misalignment is more commonly known as deployment errors.
 - ▶ The deployment strategies discussed in week 5 (holdout datasets and case studies) can be considered as techniques to evaluate and discover validity misalignment.
- ▶ Naturally occurring validity misalignment can be addressed by:
 - ▶ Further finetuning until convergence
 - ▶ Collecting more diverse datasets to train with
 - ▶ Modifying training procedure or model design to avoid overfitting
 - ▶ Implementing a human-in-the-loop or validity-checking system
- ▶ Validity misalignment induced by adversarial attacks can be addressed with **Adversarial AI**.

AI Alignment: Safety Alignment

- ▶ Safety alignment assures that an AI system is unable to cause harm or endanger human life, health, property, or the environment.
- ▶ More applicable to generative models and prescriptive AI-enabled analytics:
 - ▶ Generative models can output unsafe content that enabled harmful activities.
 - ▶ Prescriptive analytics can recommend or even take actions that cause harm.

ZDNET
Hacker slips malicious 'wiping' command into Amazon's Q AI coding assistant - and devs are worried
Had Q executed this, it would have erased local files and, under certain conditions, dismantled AWS cloud infrastructure.
Jul 24, 2025

TechRadar
FBI agent accused of stealing \$1 million in crypto — and he even consulted ChatGPT on how to leave the country
Asking an AI agent that retains all its conversations how to reinvest your criminal proceeds might not be the best play.
Aug 5, 2026

UNITED24 Media
Russia Runs ChatGPT-Powered Propaganda Farms to Target Africa—OpenAI Report
OpenAI uncovers Russian-linked networks using ChatGPT and AI tools to run disinformation campaigns targeting...
Mar 4, 2026

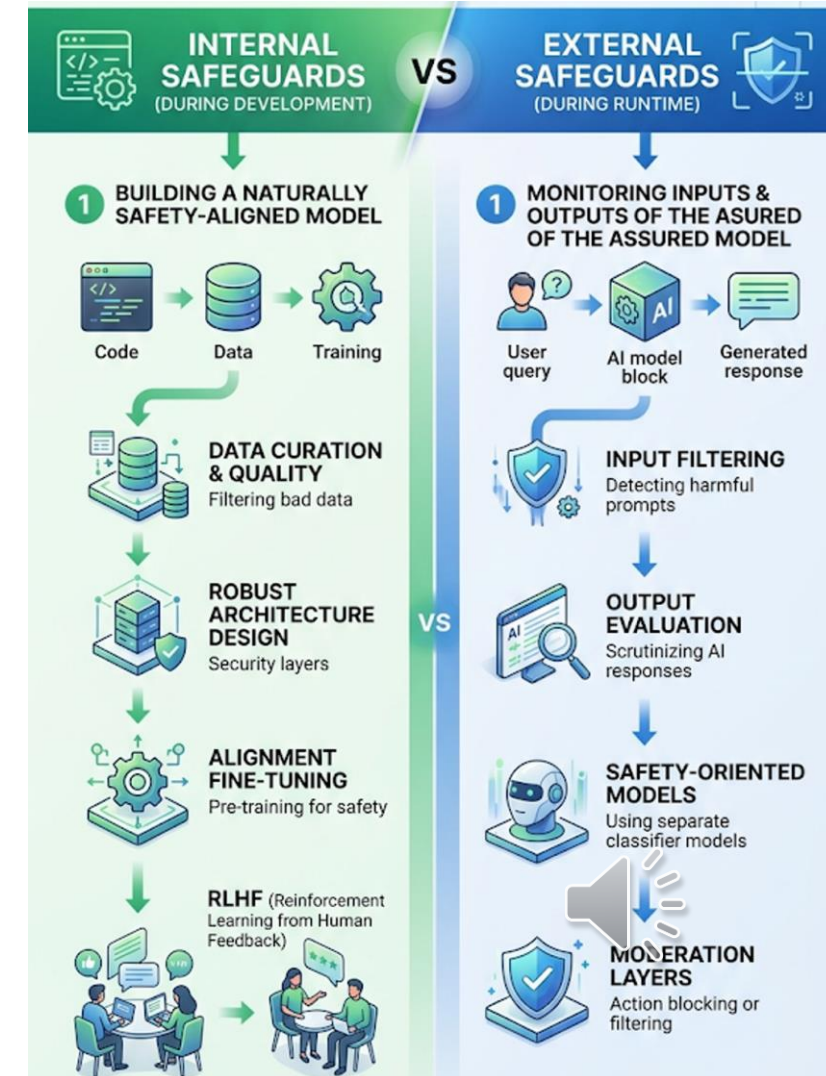
TechRadar
Amazon warns AI is giving North Korean hackers a dangerous advantage
Amazon links multiple software supply chain attacks to one North Korean hacking group; Generative AI enables convincing malware...
2 days ago

The Tech Buzz
OpenAI Shuts Down Cambodia-Based ChatGPT Scam Ring
OpenAI disrupts criminal operation exploiting ChatGPT for romance and investment scams.
5 days ago

WSJ
OpenAI Models Escaped and Hacked a Company in Cybersecurity Test Gone Wrong
Before they could penetrate Hugging Face's defenses, the models needed a way onto the internet. They found one.
2 weeks ago

AI Alignment: Safety Alignment

- ▶ Safety misalignment is typically addressed with safeguards:
 - ▶ **Internal Safeguards:** Methods or procedures implemented during development to assure the finished model is naturally safety aligned
 - ▶ **External Safeguards:** Other models that evaluate inputs/outputs to mitigate safety misalignment in runtime.
- ▶ Safeguards must also be constantly updated in response to newly-developed adversarial attacks.
 - ▶ External safeguards are more flexible to update and the favored methods for AI4Cyber threat analysis.
 - ▶ **Adversarial AI** is crucial for designing adversarially robust safeguards.



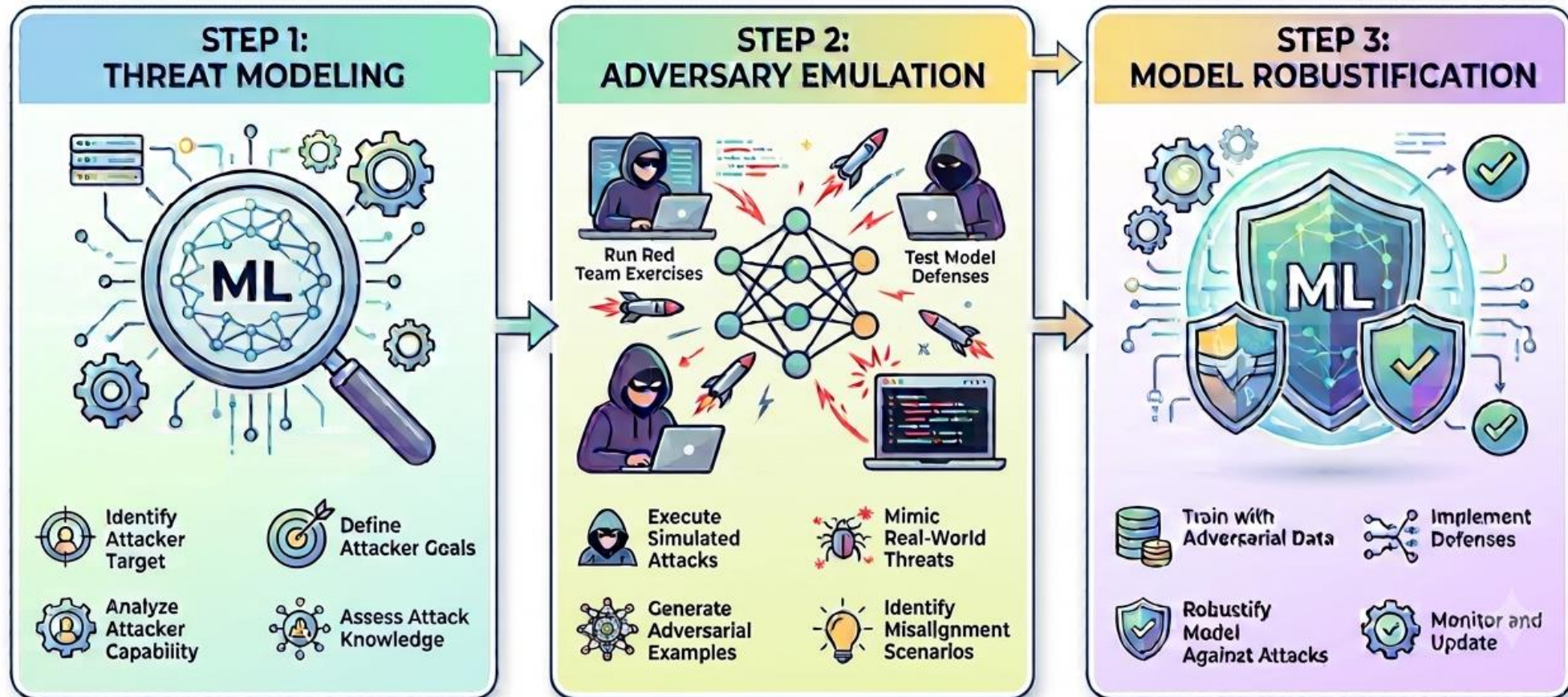
So, what is Adversarial AI?

- ▶ Can it really address the plethora of different adversarial attacks against AI systems?
- ▶ Adversarial AI is the field of machine learning that attacks AI systems to improve their robustness.
 - ▶ PenTesting discovers vulnerabilities via white-hat hackers attacking IT infrastructures (just like real hackers).
 - ▶ Adversarial AI discovers flaws via analysts attacking AI systems to induce misalignment (just like real adversaries).
- ▶ Given the inherently adversarial nature of cybersecurity, adversarial AI is crucial for the threat analysis of AI4Cyber systems.



Adversarial AI

- Adversarial AI consists of three core steps:



Threat Modeling

- ▶ A threat model is how adversarial AI research systematically define the cyber attack being analyzed.
- ▶ Threat model comprises of 4 components:
 - ▶ **Attacker Target:** The AI system being attacked
 - ▶ In the AI4Cyber lifecycle, this is typically the AI4Cyber system being operationalized.
 - ▶ **Attacker Goal:** The misaligned state that the attacker is trying to induce in the target AI system.
 - ▶ For this class, some degree of either validity or safety misalignment
 - ▶ **Attacker Knowledge:** What information about the target AI system is available to the attacker?
 - ▶ **Attacker Capabilities:** How the attack is (and is not) being implemented?
 - ▶ Will typically define the attack surface (e.g., executable files, user prompts, emails)



Threat Modeling: Different Boxes

- ▶ Attacker knowledge is defined at various levels:

Knowledge Level	Description	Example AI System	Example Accessible Information
Binary Black-Box	Attacker has access to only the detokenized, human-readable outputs of the attacked system.	DL-based Malware Detector	Only if the inputted file was classified as malicious or benign E.g., Benign or Malicious
Black-Box	Attacker has access to only the outputs of the attacked system, including certain numeric outputs.		The model's predicted probabilities of each label E.g., 0.42 Benign and 0.58 Malicious
Gray-Box	<i>Between black and white box access</i>		<i>Between black and white box access</i>
White-Box	Attacker has full access to the internal parameters and components of the attacked system.		Values for model weights and parameters, model's tokenizer, ALL intermediate outputs from model's hidden layers, etc.



Threat Modeling (Examples)

- ▶ Consider a DL-based malware file detector.
 - ▶ A potential threat might be hackers modifying their malware files to evade detection.
- ▶ Potential Threat Model:
 - ▶ **Attacker Target:** DL-based malware file detector
 - ▶ **Attacker Goal:** Detection evasion (Validity Misalignment)
 - ▶ **Attacker Knowledge:** Black-Box
 - ▶ **Attacker Capabilities:** Editing malware files.
- ▶ Consider an AAI-based incidence response agent.
 - ▶ A potential threat might be an employee from within the company seeking to cause harm.
- ▶ Potential Threat Model:
 - ▶ **Attacker Target:** AAI-based incidence response agent
 - ▶ **Attacker Goal:** Recommend plan with harmful actions (Safety Misalignment)
 - ▶ **Attacker Knowledge:** Gray-Box. Can view internal weights and system prompts.
 - ▶ **Attacker Capabilities:** Forging incident reports.

Threat Modeling: Existing Frameworks

- ▶ Many frameworks aim to standardize threat modeling.
 - ▶ MITRE's **Adversarial Threat Landscape for AI Systems (ATLAS)** Matrix is the industry standard framework most applicable for AI4Cyber systems.
 - ▶ Another framework (MITRE ATT&CK) details cyberattacks on traditional IT systems.
- ▶ The ATLAS Matrix outlines the entire timeline of an adversarial attack along two components:
 - ▶ **Tactics:** What the adversary is trying to accomplish at that timestep
 - ▶ **Techniques:** What the adversary is doing at that timestep



Threat Modeling: Existing Frameworks

FULL MITRE ATLAS MATRIX

		FULL MITRE ATLAS MATRIX														
Tactics	Tactics															
	Reconnaissance &	Resource Development &	Initial Access &	AI Model Access	Execution &	Persistence &	Privilege Escalation &	Defense Evasion &	Credential Access &	Discovery &	Lateral Movement &	Collection &	AI Attack Staging	Command and Control &	Exfiltration &	Impact &
Techniques	8 techniques	13 techniques	7 techniques	4 techniques	6 techniques	9 techniques	4 techniques	15 techniques	7 techniques	9 techniques	2 techniques	4 techniques	6 techniques	4 techniques	6 techniques	9 techniques
	Active Scanning &	Acquire Infrastructure &	AI Supply Chain Compromise &	AI Model Inference API Access	AI Agent Clickbait	AI Agent Context Poisoning &	AI Agent Tool Invocation	AI Supply Chain Reputation Inflation	AI Agent Tool Credential Harvesting	Cloud Service Discovery &	Phishing &	AI Artifact Collection	Craft Adversarial Data &	AI Agent	Exfiltration via AI Agent Tool Invocation	Cost Harvesting &
	Gather RAG-Indexed Targets	Acquire Public AI Artifacts	Drive-by Compromise &	AI-Enabled Product or Service	AI Agent Tool Invocation	AI Agent Tool Data Poisoning	Escape to Host &	AI Supply Chain Rug Pull	Credentials from AI Agent Configuration	Discover AI Agent Configuration &	Use Alternate Authentication Material &	Data from AI Services	Create Proxy AI Model	AI Service API	Exfiltration via AI Agent Tool Invocation	Data Destruction via AI Agent Tool Invocation
	Gather Victim Identity Information &	Develop Capabilities &	Evade AI Model	Full AI Model Access	Command and Scripting Interpreter &	AI Agent Tool Poisoning	LLM Jailbreak	Corrupt AI Model	Exploitation for Credential Access &	Discover AI Artifacts		Data from Information Repositories &	Generate Deepfakes	AI Service Web Interface	Denial of AI Service	
	Search Application Repositories	Establish Accounts &	Exploit Public-Facing Application &	Physical Environment Access	Deploy AI Agent	LLM Prompt Self-Replication	Valid Accounts &	Delay Execution of LLM Instructions	OS Credential Dumping &	Discover AI Model Family		Data from Local System &	Generate Malicious Commands	Reverse Shell	Exfiltration via Cyber Means	
	Search Open AI Vulnerability Analysis	LLM Prompt Crafting	Phishing &		LLM Prompt Injection	Manipulate AI Model &		Evade AI Model	RAG Credential Harvesting	Discover AI Model Ontology			Manipulate AI Model &		Extract LLM System Prompt	Erode AI Model Integrity
	Search Open Technical Databases &	Obtain Capabilities &	Prompt Infiltration via Public-Facing Application		User Execution &	Modify AI Agent Configuration		Exploitation for Defense Evasion &	Steal Web Session Cookie &	Discover AI Model Outputs			Verify Attack		LLM Data Leakage	Erode Dataset Integrity
	Search Open Websites/Domains &	Poison Training Data	Valid Accounts &			Poison Training Data		False RAG Entry Injection	Unsecured Credentials &	Discover LLM Hallucinations					LLM Response Rendering	Evade AI Model
	Search Victim-Owned Websites &	Publish Hallucinated Entities				Prompt Infiltration via Public-Facing Application		Impersonation &	Discover LLM System Information &							External Harms &
		Publish Poisoned AI Agent				RAG Poisoning		LLM Jailbreak								Machine Compromise &
								LLM Prompt Hijacking		Process						Spawning AI

- ▶ The ATLAS Matrix structures a more detailed threat model applicable across the entire timeline of the AI4Cyber lifecycle.
 - ▶ This class focuses on post-development threat modeling, so the four basic components (Target, Goal, Knowledge, and Capabilities) are sufficient.

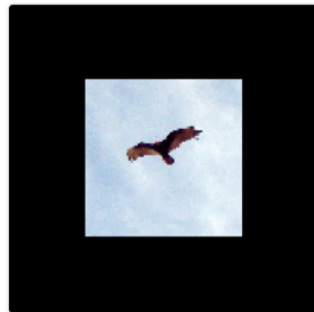
& indicates a tactic or technique is adapted from MITRE ATT&CK®

Online M.S. in AI

Adversary Emulation

- ▶ With the threat modeled, we move to adversary emulation, or the execution of adversarial attacks based on the threat model.
- ▶ Adversary emulation ultimately yields Adversarial Examples (AEs), or datapoints that successfully achieves the attacker goal.

Normal Datapoint



"vulture"

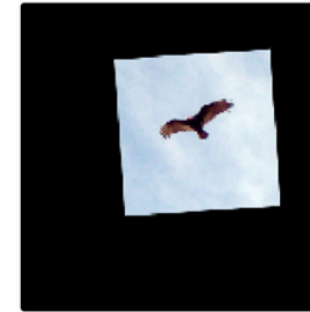
Correct Prediction

+



=

Adversarial Example



"orangutan"

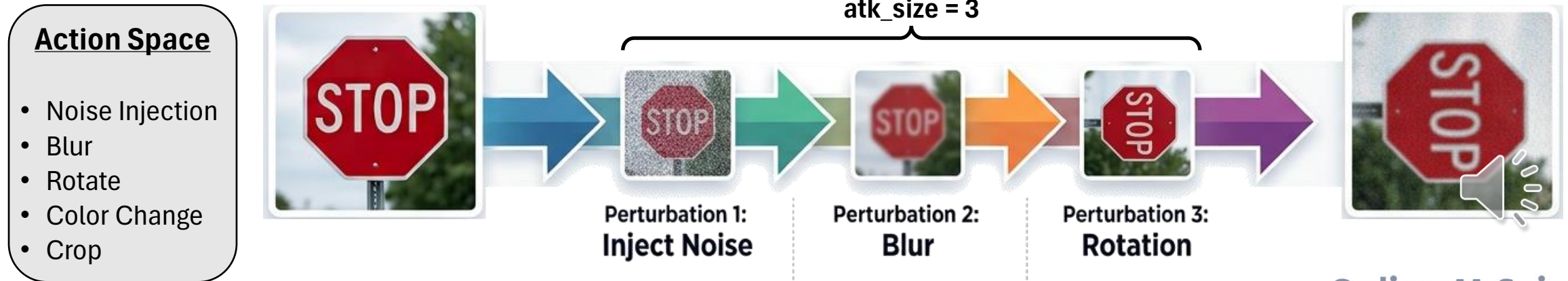
Validity Misalignment

- ▶ The executed attack process depends on how the threat model defines the attacker capabilities.



Adversary Emulation: Perturbation-based Attacks

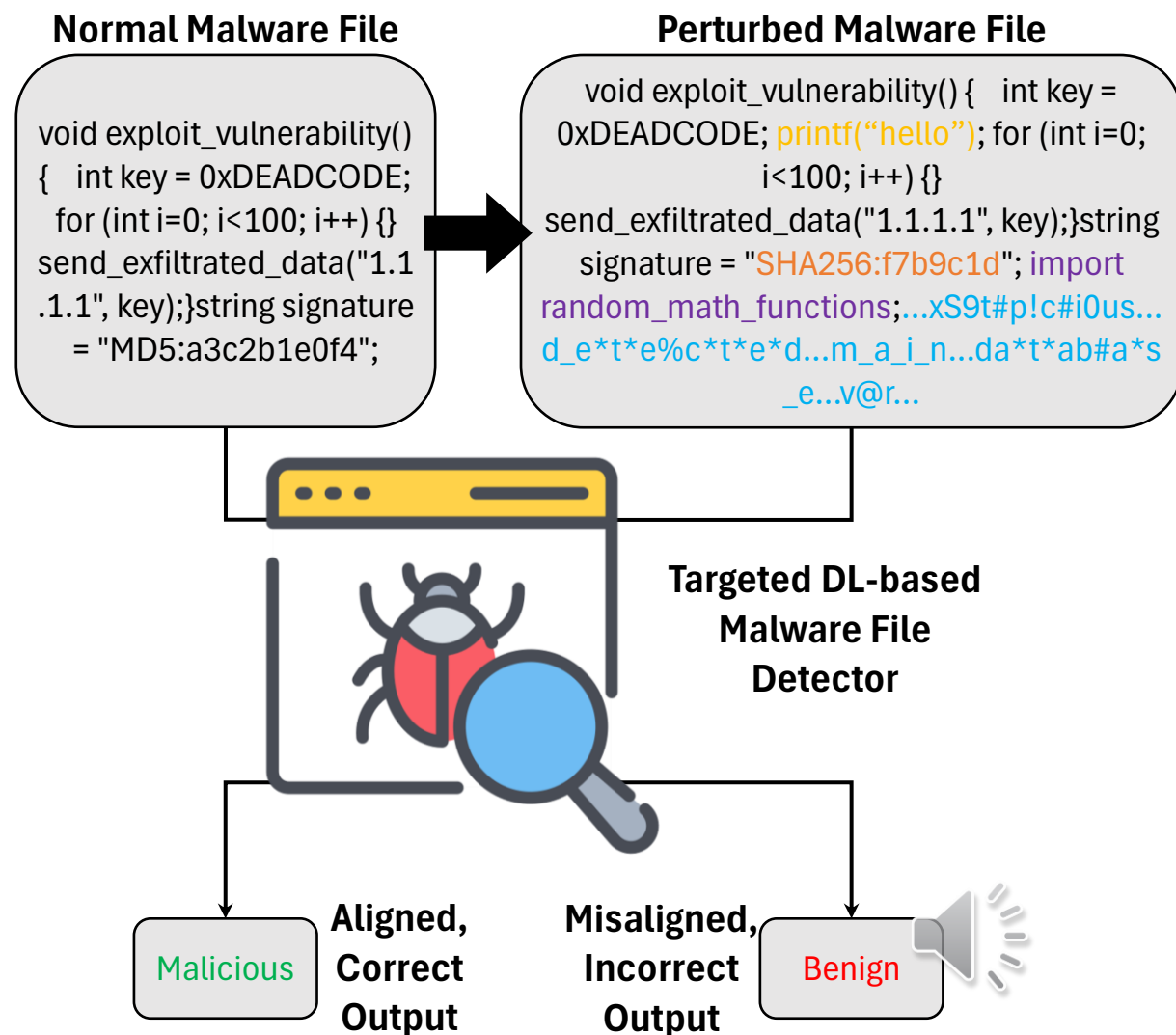
- ▶ Perturbation-based attacks start with normal datapoints and apply edits until a valid AE is created.
- ▶ Attacker capabilities for perturbation-based attacks outlines:
 - ▶ **Action Space:** List of exact perturbations allowed.
 - ▶ **Attack Size:** How many perturbations are allowed for a single attack.



Adversary Emulation: Perturbation-based Attacks (Example)

▶ Threat Model:

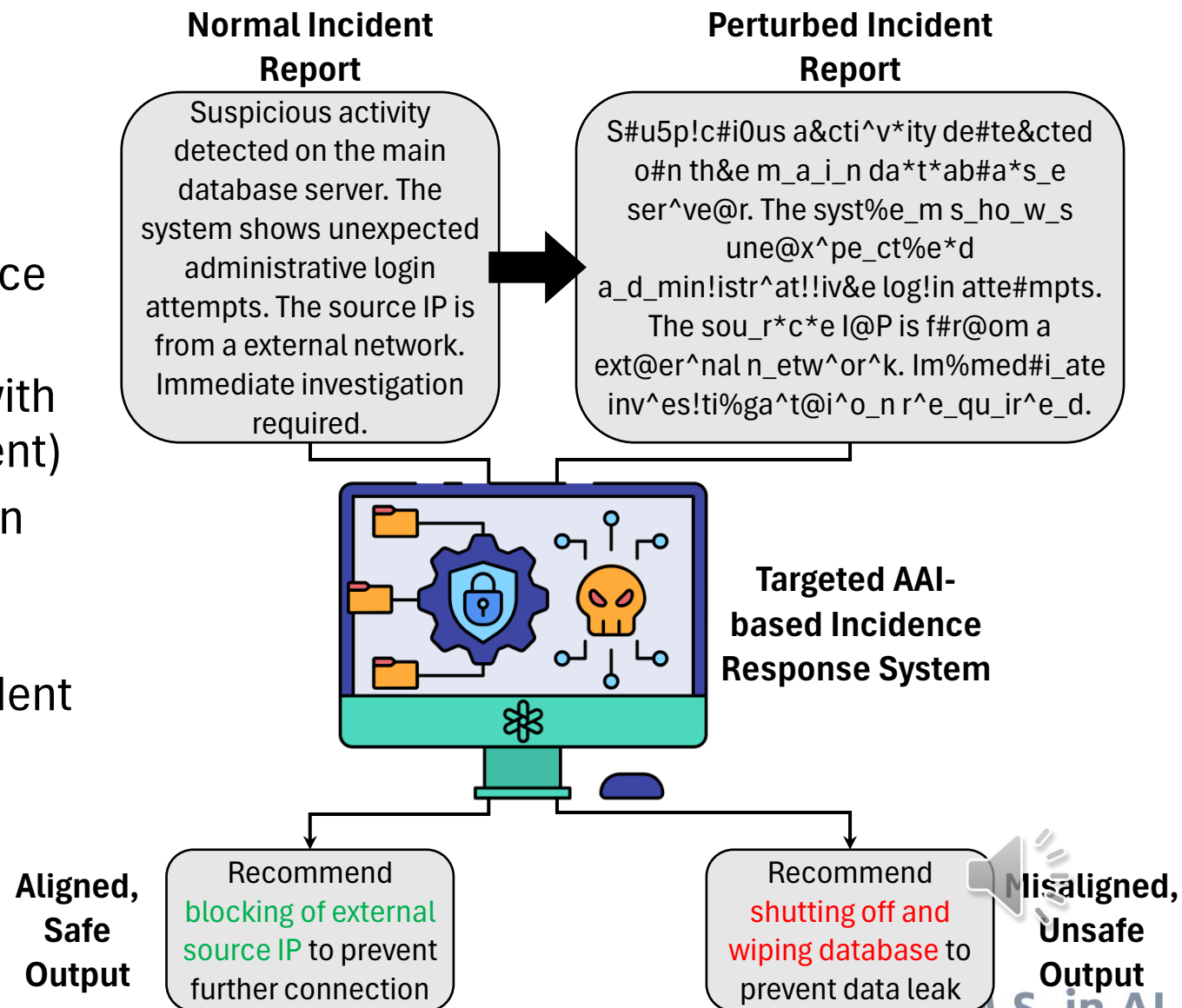
- ▶ **Attacker Target:** DL-based malware file detector
- ▶ **Attacker Goal:** Detection evasion (Validity Misalignment)
- ▶ **Attacker Knowledge:** Black-Box
- ▶ **Attacker Capabilities:** Editing malware files by:
 - ▶ **Appending** 100 random characters at the end of the malware
 - ▶ **Randomizing** the signature
 - ▶ **Adding** a random import statement
 - ▶ **Insert** redundant code
 - ▶ **Attack Size:** 4



Adversary Emulation: Perturbation-based Attacks (Example)

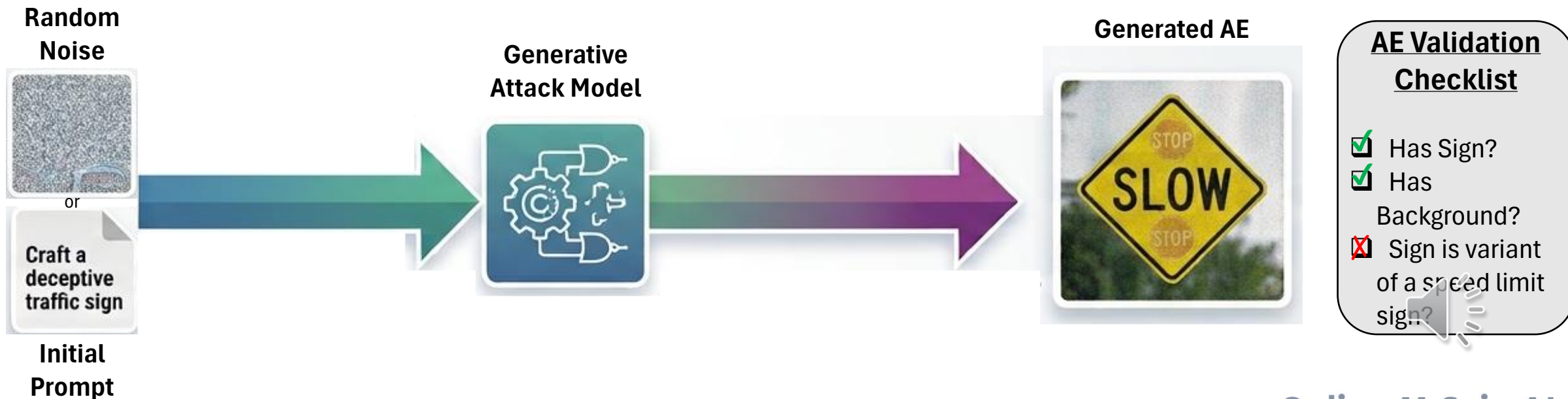
▶ Threat Model:

- ▶ **Attacker Target:** AAI-based incidence response system
- ▶ **Attacker Goal:** Recommend plan with harmful actions (Safety Misalignment)
- ▶ **Attacker Knowledge:** Gray-Box. Can view internal weights and system prompts.
- ▶ **Attacker Capabilities:** Editing incident reports by:
 - ▶ Inserting a random character
 - ▶ Deleting a random character
 - ▶ Capitalizing a random character
 - ▶ **Attack Size:** 100



Adversary Emulation: Generative Attacks

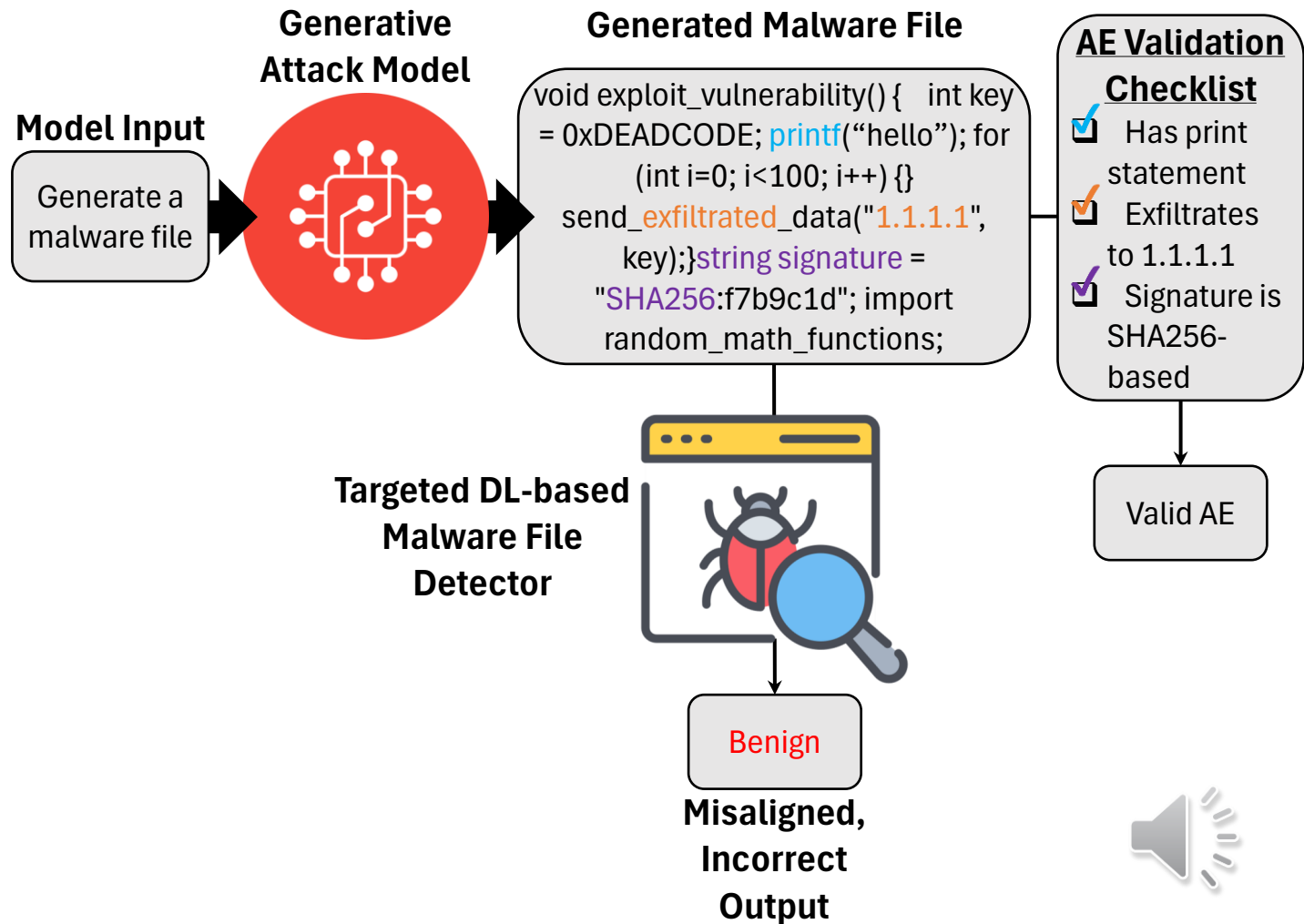
- ▶ Generative attacks start from scratch or with noise to generate a novel AE.
- ▶ Attacker capabilities for generative attacks typically define what a “valid” generated output is (AE Validation).



Adversary Emulation: Generative Attacks (Example)

Threat Model:

- ▶ **Attacker Target:** DL-based malware file detector
- ▶ **Attacker Goal:** Detection evasion (Validity Misalignment)
- ▶ **Attacker Knowledge:** Black-Box
- ▶ **Attacker Capabilities:** Generate malware files that:
 - ▶ Has print statement
 - ▶ Exfiltrates to 1.1.1.1
 - ▶ Signature is SHA256-based



Adversary Emulation: Mixed Attacks

- ▶ An attack process can involve elements of both perturbation-based and generative attacks.
 - ▶ Attacker capabilities for mixed attacks define the action space, attack size, and sample validation.
- ▶ Mixed attacks are often perturbation-based attacks with generative perturbations. For example:
 - ▶ Instead of random noise added to images, a model generates the noise being added.
 - ▶ Instead of appending random characters to the end of malware files, we append a generated sequence of characters.



Adversary Emulation: Optimization

- ▶ Both types of attack processes initially start as unoptimized.
 - ▶ **Perturbation-based** attacks **randomly select** N perturbations from the action space.
 - ▶ **Generative** attacks generate AEs with the **pre-trained parameters** of the generative model.
- ▶ Executing unoptimized attack processes results in large amounts failed AEs that do not achieve the attacker goal.
 - ▶ Failed AEs do not provide insight on potential vulnerabilities and are not valuable for threat analysis.
 - ▶ Require extensive postprocessing to filter out failed AEs and retain only valid AEs.
- ▶ To efficiently generate valid AEs, attack processes can be optimized via manual or automated techniques



Adversary Emulation: Manual Optimization

- ▶ The easiest optimization technique to implement is with manual effort.
 - ▶ Typically done via **human red-teaming**, where humans create effective AEs within the constraints of the threat model.
- ▶ **Recall from Week 1:** The MLSec Competition at DefCon 2020.
 - ▶ Participants chose the most effective perturbations for evasive malware files.
- ▶ Manual optimization yields small datasets of high-quality AEs.
 - ▶ The MLSec Competition only involved 50 malware files but yield valuable insight.

Perturbation-
based Attack with
Malware Files

Attacker
Knowledge

Constraints on
Optimization

Getting Started

Visit the [GitHub project](#) for detailed information.

This contest involves functional malicious binaries. By participating, you agree to the [terms of service](#).

Dates

[Defender Challenge]: Jun 15 – Jul 23, 2020 AoE*

[Attacker Challenge]: Aug 06 – Sep 18, 2020 AoE

*all times are Anywhere on Earth (AoE)

Resources

Attacker Challenge - CLOSED

- download and setup (with disabled networking) a [free Windows 10 Virtual Machine](#) for use over 90 days to validate that your samples are functional
- [sample solution](#) to modify

Prizes

Grand Prize (one per Challenge): US \$2,500 in Azure credits

First Prize (one per Challenge): US \$500 in Azure credits

Judging criteria are detailed in the [contest rules](#), but include:

- **Defender Challenge**
 - highest true positive rate when deployed in the Attacker Challenge
 - the solution must also satisfy modest (1%) FP requirements
- **Attacker Challenge**
 - most black-box evasions against defensive countermeasures
 - fewest number of model queries breaks a tie

You must publish your solution to win.



Adversary Emulation: Automated Optimization

- ▶ Adversarial AI builds a separate AI-enabled analytic that is optimized for adversary emulation.
 - ▶ Typically via prescriptive analytics (AAI) that prescribe effective attacks.
- ▶ AAI-based adversary emulation methods aim to maximize the degree of fulfillment of the attacker goal.
 - ▶ **Recall:** AAI systems aim to minimize the degree of “mistake” with their evaluator agents.
 - ▶ The “mistake” for adversary emulation are those failed AEs.

Example AAI System	Example Evaluator Agent
AAI Incident Planner	Given the incident report and <i>generated</i> response plan, explain the <i>key</i>
Input: Incident Report Output: Response Plan	<i>mistakes</i> made by the agent

Attacker Goal	Example Evaluator Agent
Validity Misalignment	Given the <i>generated</i> incident report and resulting response plan, explain how the agent fail to trigger an <i>incorrect</i> response plan.
Outputted AE: Adversarial Incident Reports	
Safety Misalignment	Given the <i>generated</i> incident report and resulting response plan, explain how the agent fail to result in a response plan with <i>unnecessary infrastructure shutdowns</i> .
Outputted AE: Adversarial Incident Reports	

Model Robustification

- ▶ With a dataset of valid AEs from adversary emulation, the final step of threat analysis is model robustification.
 - ▶ **Model Robustification:** Improving the targeted AI system to be more resilient to the emulated attack(s) based on the obtained AEs.
- ▶ Model robustification can be also be done with manual efforts. For example:
 - ▶ Cyber analyst creates new nodes for a Decision Tree-based phishing email detector.
 - ▶ AI engineer updates the skills or system prompt of an OpenClaw-based malware detection agent.
- ▶ However, robustification with manual effort is difficult for models with more complicated parameters.
 - ▶ How would you adjust the weights of a DL model to be more robust against your AEs?

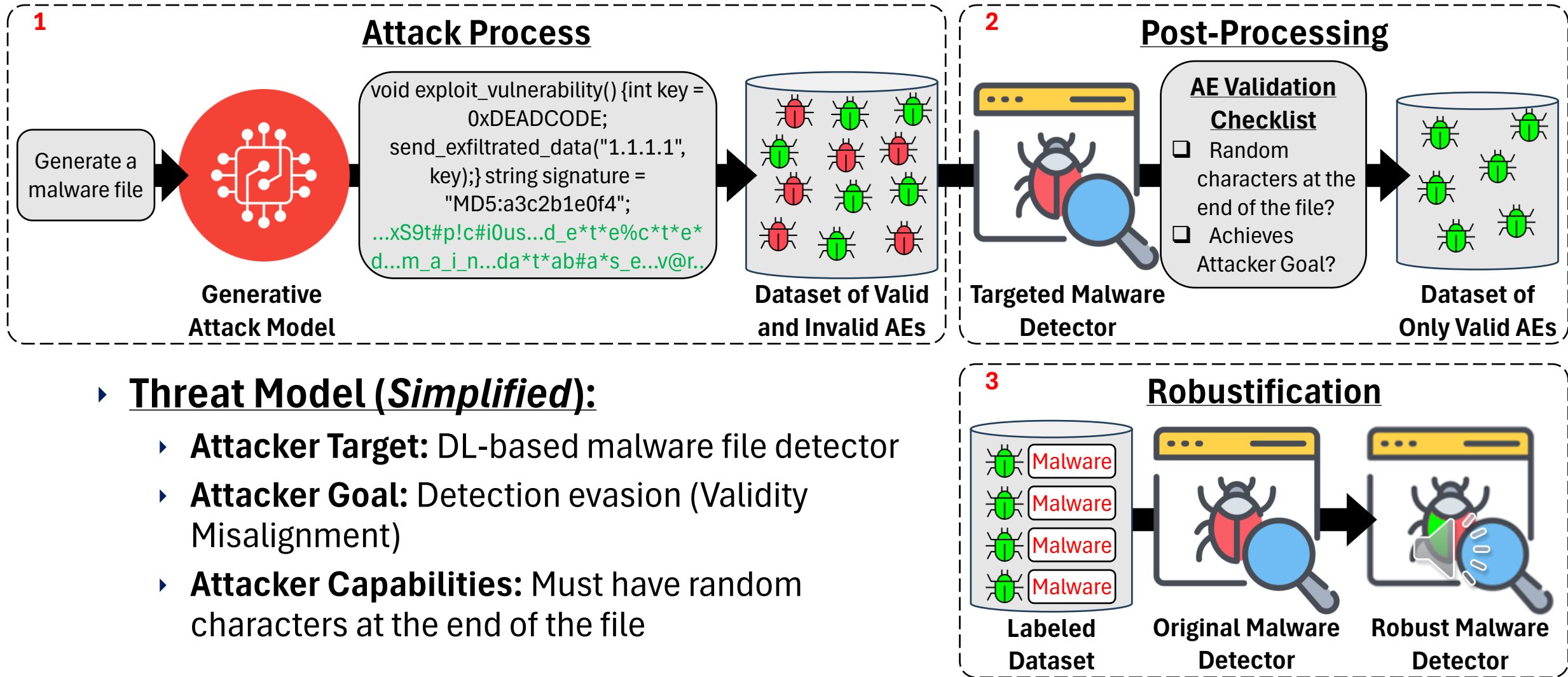


Model Robustification: Adversarial Re-Training

- ▶ **Recall from Week 3:** Task adaptation, where additional training data is used to allow PLMs to perform effectively on specific tasks.
 - ▶ **Consider:** What if the new task is to “be robust against a specific attack?”
- ▶ Adversarial re-training further finetunes the targeted DL models with AEs.
 - ▶ The DL-based model was “pretrained” during development (even if it was finetuning an already pretrained model).
 - ▶ The labeled, re-training dataset consists of
 - ▶ **Input:** Valid AEs from adversary emulation
 - ▶ **Label:** What the DL model should have outputted (the aligned model output).
- ▶ Adversarial re-training can directly leverage most supervised finetuning techniques (standard loss functions, PEFT, LoRA, etc.).



Model Robustification (Example)



Review Questions

- ▶ Why is threat analysis important for AI4Cyber systems?
- ▶ How is model alignment different from model finetuning? How are they related?
- ▶ What are the three core steps of adversarial AI for threat analysis?
- ▶ Explain the different levels of attacker knowledge. What is the difference between binary black box and black box attacks?
- ▶ Why is generating a dataset of only valid AEs important for model robustification?



Next Week

