Machine Learning Techniques in Defensive and Offensive Cybersecurity

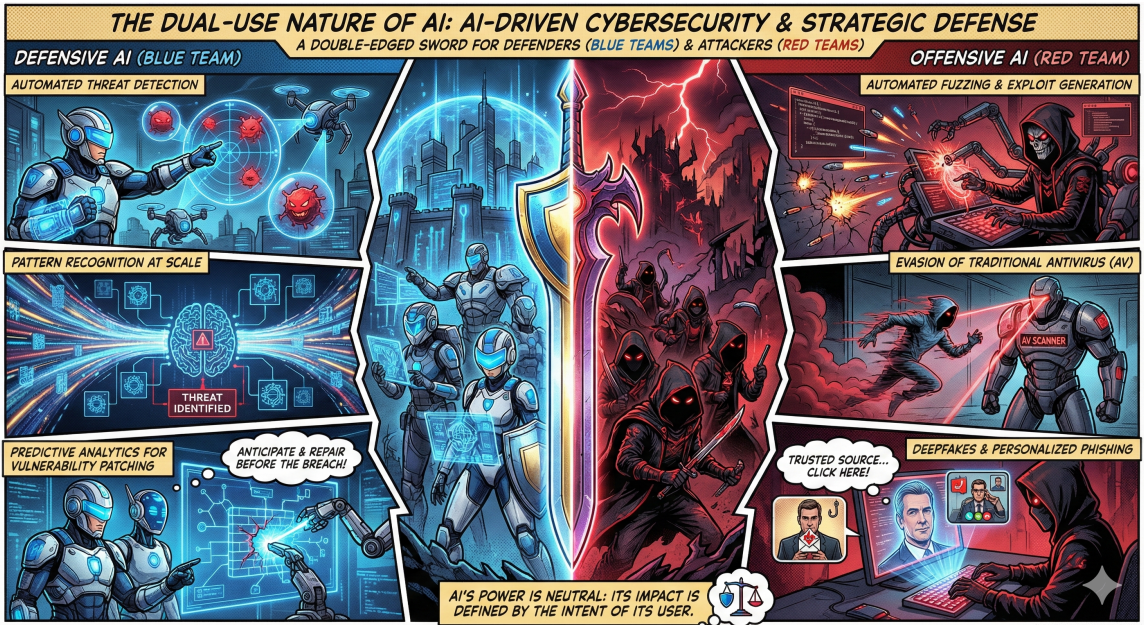
From Supervised Learning to Multi-Agent RL

Christoph Würsch, ICE, OST

December 8, 2025

Agenda

1. Introduction to AI in Cybersecurity
2. Supervised Learning
   - Classification Examples
   - Regression Examples
3. Unsupervised Learning
4. Reinforcement Learning (RL)
5. Summary
6. Generative AI & LLMs
   - Offensive GenAI
   - Defensive GenAI


THE DUAL-USE NATURE OF AI: AI-DRIVEN CYBERSECURITY & STRATEGIC DEFENSE
A DOUBLE-EDGED SWORD FOR DEFENDERS (BLUE TEAMS) & ATTACKERS (RED TEAMS)
DEFENSIVE AI (BLUE TEAM)
OFFENSIVE AI (RED TEAM)
AUTOMATED THREAT DETECTION
AUTOMATED FUZZING & EXPLOIT GENERATION
PATTERN RECOGNITION AT SCALE
THREAT IDENTIFIED
EVASION OF TRADITIONAL ANTIVIRUS (AV)
AV SCANNER
PREDICTIVE ANALYTICS FOR VULNERABILITY PATCHING
ANTICIPATE & REPAIR BEFORE THE BREACH!
DEEPFAKES & PERSONALIZED PHISHING
TRUSTED SOURCE... CLICK HERE!
AI'S POWER IS NEUTRAL: ITS IMPACT IS DEFINED BY THE INTENT OF ITS USER.

The Dual-Use Nature of AI

Artificial Intelligence in cybersecurity is a double-edged sword, utilized by both defenders (Blue Teams) and attackers (Red Teams).

Defensive AI (Blue Team)

- Automated threat detection.
- Pattern recognition at scale.
- Predictive analytics for vulnerability patching.

Offensive AI (Red Team)

- Automated fuzzing and exploit generation.
- Evasion of traditional antivirus (AV).
- Deepfakes and personalized phishing.

Why AI is Critical: Statistics

Traditional signature-based detection is no longer sufficient.

- **Volume:** AV-TEST registers over **450,000** new malicious programs every day.
- **Cost:** According to IBM's *Cost of a Data Breach Report 2023*, the global average cost of a data breach is **$4.45 million**.
- **Speed:** AI-driven security automation reduces the breach lifecycle by an average of **108 days**.
- **Complexity:** "Fileless" malware and "Living off the Land" attacks require behavioral analysis, not just file scanning.

Malware

Malware

Malware is short for malicious software. It refers to any software that is intentionally designed to harm, disrupt, or gain unauthorized access to a computer, network, or data.
Goal: Theft, damage, spying, extortion, or unauthorized control.

- **Virus** – attaches to files and spreads when those files are opened
- **Worm** – spreads automatically across networks
- **Trojan** – disguised as legitimate software
- **Ransomware** – locks your files and demands payment
- **Spyware** / keyloggers – secretly collect information

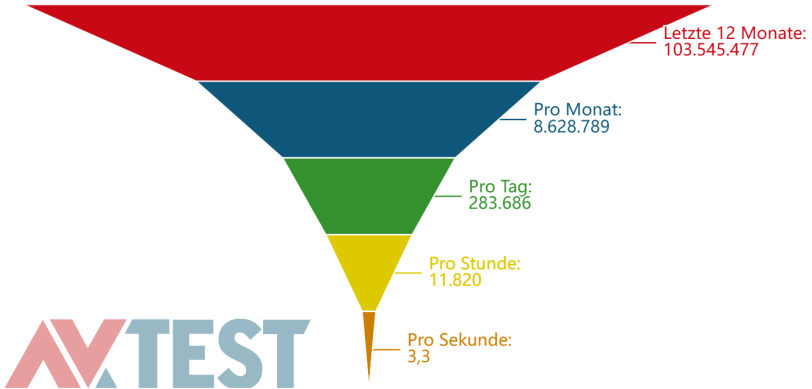
PUA: Potentially Unwanted Application

PUA

A **Potentially Unwanted Application (PUA)** is software that is not necessarily malicious, but is generally undesirable because it may:

- Show ads or pop-ups (adware)
- Install extra programs you didn't ask for
- Slow down your system
- Change browser settings (search engine, homepage)
- Track your browsing for marketing

Malware and PUA per second



Total amount of malware and PUA (potentially unwanted application)

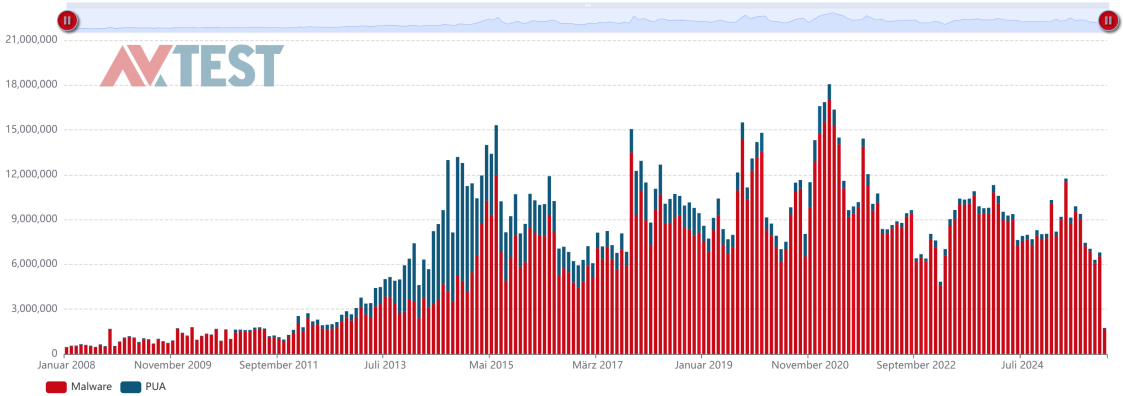
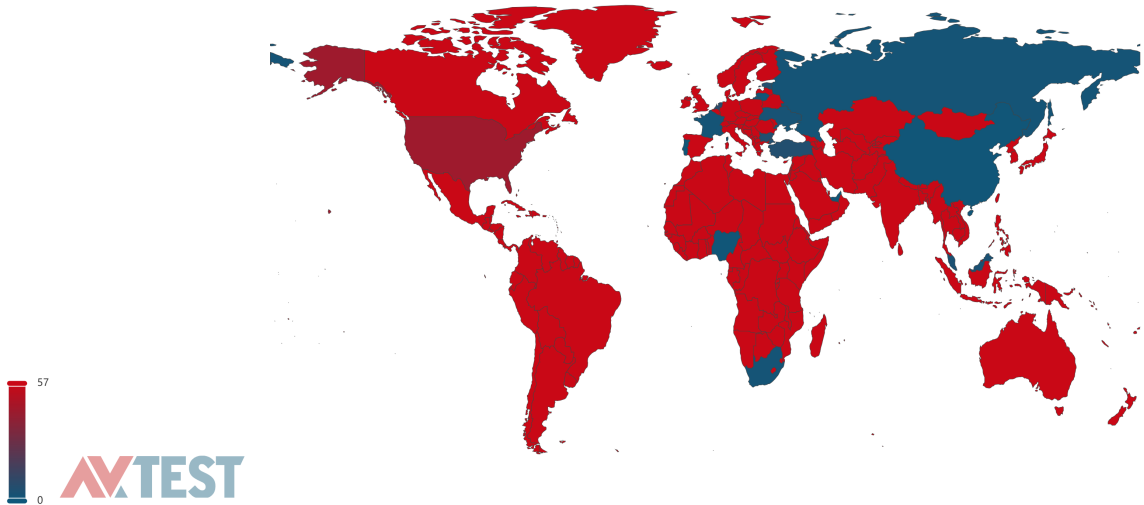


Figure: Total amount of malware and pua.

Origin of malicious SPAM



Operating Systems

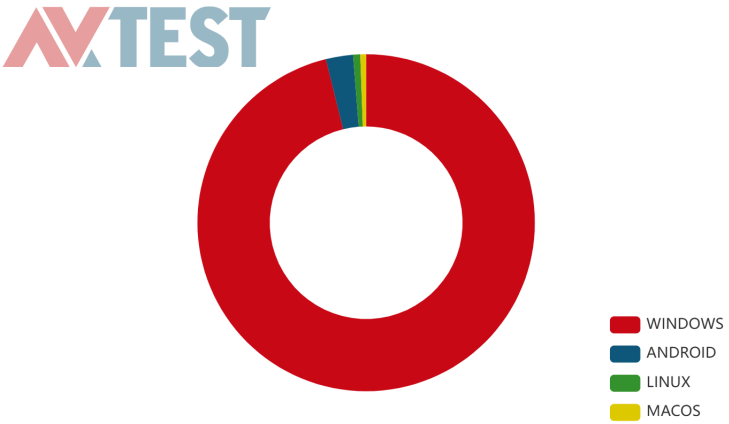


Figure: Distribution of malware with respect to operating system.

Real-World Case Studies

1. **Offensive: DeepLocker (IBM Research)**
   - A "stealthy" malware proof-of-concept.
   - Uses AI to hide its payload within a benign video conferencing app.
   - Only executes the attack when it recognizes the target's face via the webcam (Facial Recognition triggering).

2. **Defensive: Darktrace (The Enterprise Immune System)**
   - Uses unsupervised learning to learn the "pattern of life" for every device in a network.
   - Detected the "WannaCry" ransomware attack in real-time by noticing anomalous SMB traffic, stopping it before encryption spread.

Supervised Learning Overview

Definition: Training a model on a labeled dataset (Input $\rightarrow$ Output).
Primary Use: Detecting known threat patterns.

Two main sub-categories:

- **Classification:** Predicting a category (e.g., Malicious vs. Benign).
- **Regression:** Predicting a continuous value (e.g., Risk Score).

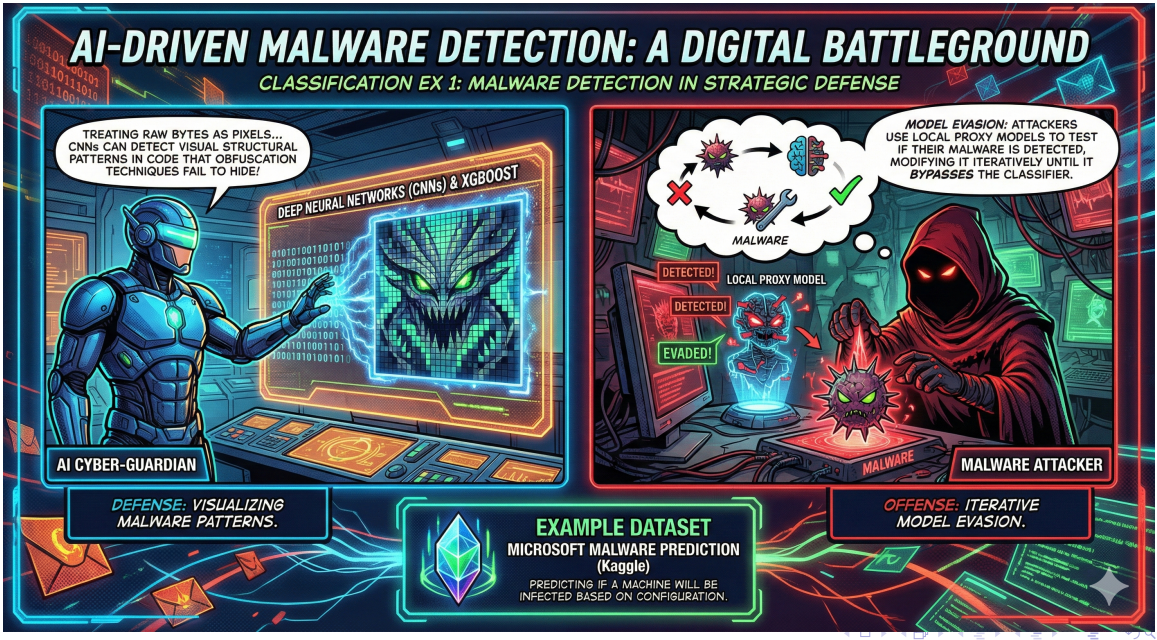
Classification Ex 1: Malware Detection

Technique: Deep Neural Networks (CNNs) or XGBoost.

- **Defense:** Treating raw bytes of a file as pixels in an image. CNNs can detect visual structural patterns in code that obfuscation techniques fail to hide.
- **Offense:** *Model Evasion.* Attackers use local proxy models to test if their malware is detected, modifying it iteratively until it bypasses the classifier.

Example Dataset

Dataset: Microsoft Malware Prediction (Kaggle)
Predicting if a machine will be infected based on configuration.


AI-DRIVEN MALWARE DETECTION: A DIGITAL BATTLEGROUND
CLASSIFICATION EX 1: MALWARE DETECTION IN STRATEGIC DEFENSE
TREATING RAW BYTES AS PIXELS... CNNs CAN DETECT VISUAL STRUCTURAL PATTERNS IN CODE THAT OBFUSCATION TECHNIQUES FAIL TO HIDE!
DEEP NEURAL NETWORKS (CNNs) & XGBOOST
AI CYBER-GUARDIAN
MODEL EVASION: ATTACKERS USE LOCAL PROXY MODELS TO TEST IF THEIR MALWARE IS DETECTED, MODIFYING IT ITERATIVELY UNTIL IT BYPASSES THE CLASSIFIER.
MALWARE
DETECTED!
LOCAL PROXY MODEL
DETECTED!
EVADED!
MALWARE
MALWARE ATTACKER
DEFENSE: VISUALIZING MALWARE PATTERNS.
EXAMPLE DATASET
MICROSOFT MALWARE PREDICTION (Kaggle)
PREDICTING IF A MACHINE WILL BE INFECTED BASED ON CONFIGURATION.
OFFENSE: ITERATIVE MODEL EVASION.

Classification Ex 2: Phishing & Spam Detection

Technique: Natural Language Processing (NLP) with SVMs or BERT.

- **Defense:** Analyzes email headers, URL structures, and semantic context (urgency, financial requests) to classify emails as "Ham" or "Spam".
- **Offense:** *Spear-Phishing.* Attackers train models on a target's social media history to generate emails that mimic the writing style of colleagues.

Example Dataset

Dataset: Enron Email Dataset (Kaggle)
Large corpus of real corporate emails for training NLP models.

Regression Ex 1: Vulnerability Scoring

Technique: Linear Regression or Random Forest Regressor.

- **Concept:** Predicting the Common Vulnerability Scoring System (CVSS) score (0.0 to 10.0) before an official analysis is released.
- **Utility:** Helps security teams prioritize which patch to apply immediately based on predicted severity.

Example Dataset

Dataset: NIST NVD Data Feeds (JSON/CSV)
Official US government repository of vulnerabilities.

Regression Ex 2: Network Traffic Forecasting

Technique: Long Short-Term Memory (LSTM) RNNs.

- **Defense:** Predicts expected traffic volume for the next time window.
- **Formula:** If Actual $>$ (Predicted $+$ Threshold), trigger DDoS alert.
- **Offense:** Attackers use regression to determine the minimum traffic spike required to crash a server without triggering volume-based firewalls.

Example Dataset

Dataset: CIC-IDS2017 (UNB)
Comprehensive dataset including DDoS, Heartbleed, and Brute Force attacks.

Unsupervised Learning Overview

Definition: Finding hidden patterns in unlabeled data.
Primary Use: Zero-Day detection and Insider Threat detection.
The system does not know what an attack looks like; it only knows what "normal" looks like.

Unsupervised Ex 1: Network Anomaly Detection

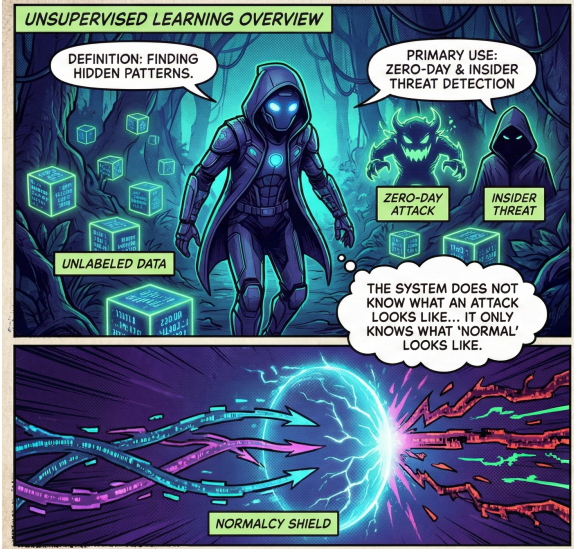
Technique: Isolation Forests or Autoencoders.

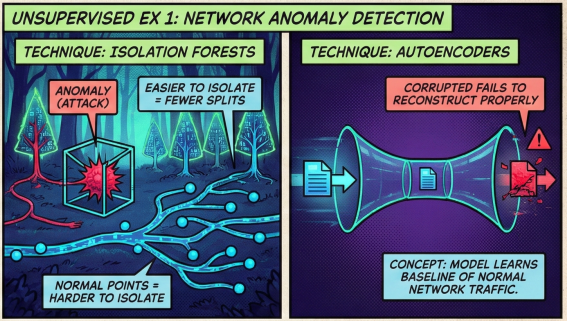
- **Concept:** The model learns a baseline of normal network traffic.
- **Detection:** "Isolation Forests" isolate anomalies by randomly selecting a feature and splitting values. Anomalies (attacks) are easier to isolate (require fewer splits) than normal points.
- **Application:** Detecting data exfiltration (large uploads) at unusual hours.

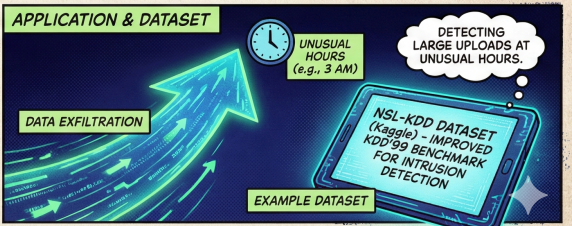
Example Dataset

Dataset: NSL-KDD Dataset (Kaggle)
Improved version of the KDD'99 benchmark for intrusion detection.

ANOMALY DETECTION IN AI-DRIVEN CYBERSECURITY: UNSUPERVISED LEARNING







Unsupervised Ex 2: User Behavior Analytics (UEBA)

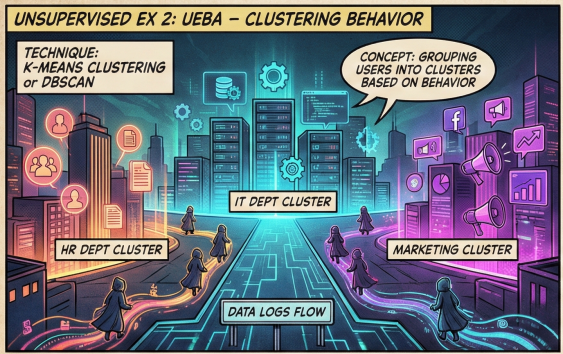
Technique: K-Means Clustering or DBSCAN.

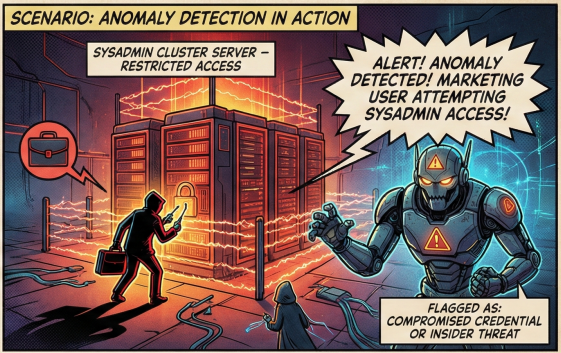
- **Concept:** Grouping users into clusters based on behavior (e.g., HR dept, IT dept).
- **Scenario:** If a user typically in the "Marketing Cluster" attempts to access a server usually only touched by the "SysAdmin Cluster," it is flagged as a compromised credential or insider threat.

Example Dataset

Dataset: CERT Insider Threat Dataset (CMU)
Synthetic dataset containing logs of insider threats.

AI-DRIVEN CYBERSECURITY & STRATEGIC DEFENSE: USER BEHAVIOR ANALYTICS (UEBA)





Semi-Supervised Ex 1: Password Cracking (GANs)

Technique: Generative Adversarial Networks (PassGAN).

- **Mechanism:**
  - **Generator:** Creates fake passwords.
  - **Discriminator:** Tries to distinguish real human passwords (from leaks) vs. AI-generated ones.
- **Outcome:** The Generator learns the underlying probability distribution of human password choices, generating guesses that are far more effective than random brute force.

Example Dataset

Dataset: RockYou.txt (SecLists)
The gold standard wordlist for password cracking research.

Semi-Supervised Ex 2: Malicious Domain Labeling

Technique: Label Propagation (Graph-based).

- **Problem:** We know 100 bad domains, but have 1 million unknown domains.
- **Method:** Construct a graph where nodes are domains and edges are shared attributes (same IP, same registrar, same DNS nameserver).
- **Action:** Propagate the "Malicious" label to unknown nodes that are heavily connected to known bad nodes.

Example Dataset

Dataset: Malicious URLs Dataset (Kaggle)
Contains URLs labeled as benign, defacement, phishing, and malware.

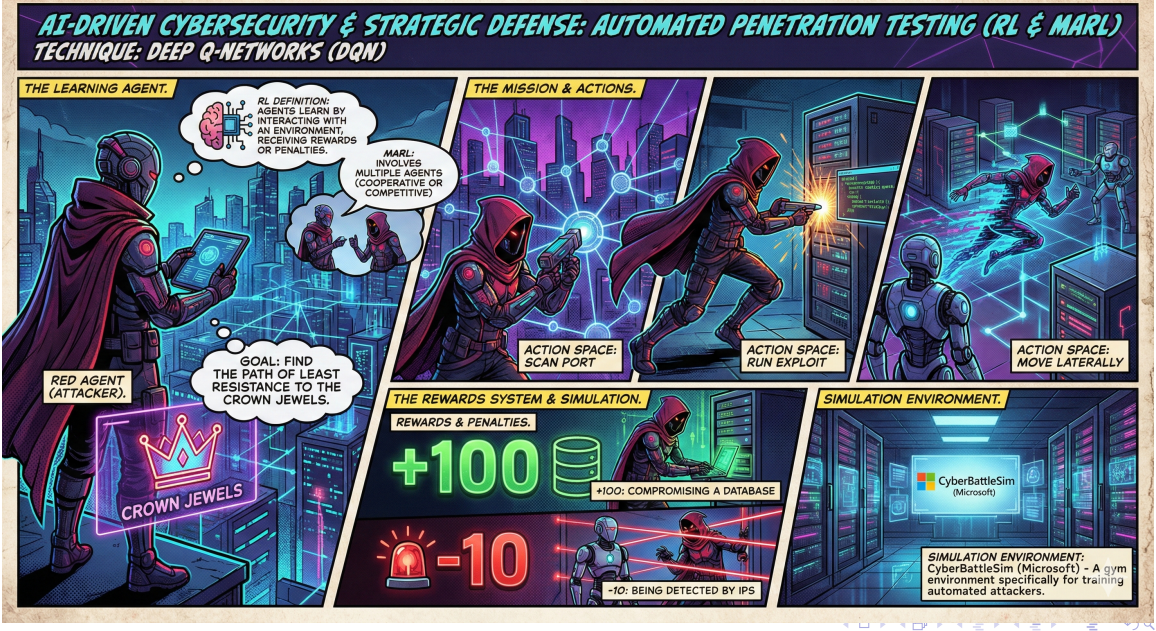
RL Ex 1: Automated Penetration Testing

Definition: Agents learn by interacting with an environment, receiving rewards or penalties.
MARL (Multi-Agent RL): Involves multiple agents (cooperative or competitive).
Technique: Deep Q-Networks (DQN).

- **Role:** Red Agent (Attacker).
- **Action Space:** Scan Port, Run Exploit, Move Laterally.
- **Rewards:**
  - +100: Compromising a database.
  - -10: Being detected by IPS.
- **Goal:** Find the path of least resistance to the Crown Jewels.

Simulation Environment

Dataset: CyberBattleSim (Microsoft)
A gym environment specifically for training automated attackers.


AI-DRIVEN CYBERSECURITY & STRATEGIC DEFENSE: AUTOMATED PENETRATION TESTING (RL & MARL)
TECHNIQUE: DEEP Q-NETWORKS (DQN)
THE LEARNING AGENT.
RL DEFINITION: AGENTS LEARN BY INTERACTING WITH AN ENVIRONMENT, RECEIVING REWARDS OR PENALTIES.
MARL: INVOLVES MULTIPLE AGENTS (COOPERATIVE OR COMPETITIVE)
GOAL: FIND THE PATH OF LEAST RESISTANCE TO THE CROWN JEWELS.
RED AGENT (ATTACKER).
CROWN JEWELS
THE MISSION & ACTIONS.
ACTION SPACE: SCAN PORT
ACTION SPACE: RUN EXPLOIT
ACTION SPACE: MOVE LATERALLY
THE REWARDS SYSTEM & SIMULATION.
REWARDS & PENALTIES.
+100
+100: COMPROMISING A DATABASE
-10
-10: BEING DETECTED BY IPS
SIMULATION ENVIRONMENT.
CyberBattleSim (Microsoft)
SIMULATION ENVIRONMENT: CyberBattleSim (Microsoft) - A gym environment specifically for training automated attackers.

Operation Fortitude — Allied Deception Before D-Day

- **Goal:** Mislead Germany about the location and timing of the Allied invasion.
- **Components:**
  - *Fortitude North:* Simulated invasion threat against Norway.
  - *Fortitude South:* Fake main invasion aimed at Pas-de-Calais.
- **Tactics:**
  - Phantom army (FUSAG) under General Patton.
  - Inflatable tanks, dummy aircraft, fake landing craft.
  - Simulated radio traffic, staged troop movements.
  - Double agents feeding false intelligence.
- **Effect:** German forces held in wrong locations, aiding success of D-Day.

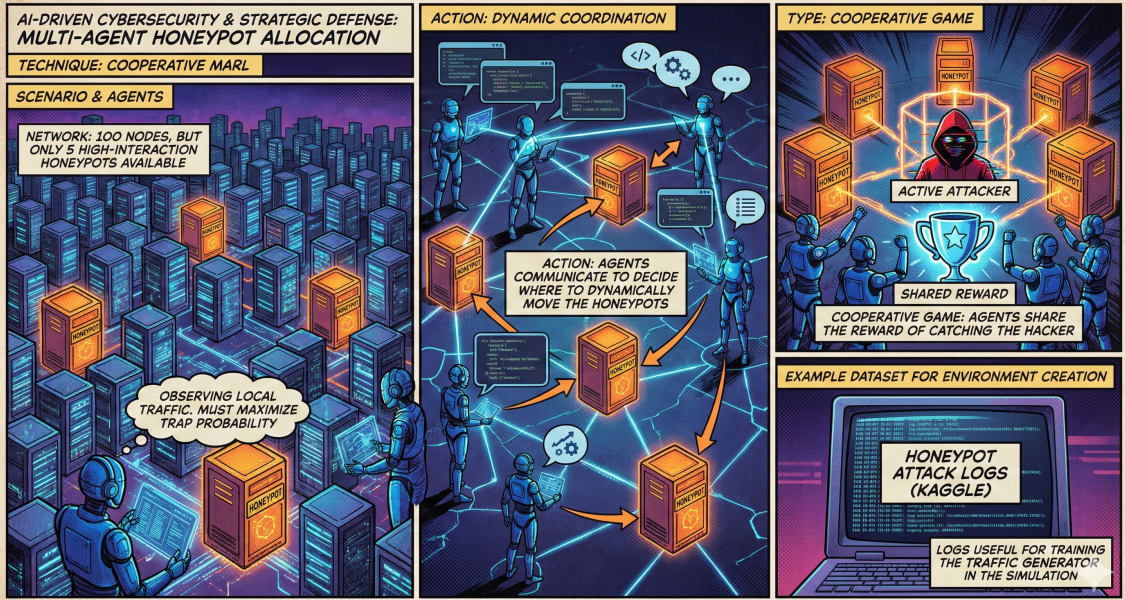
RL Ex 2: Multi-Agent Honeypot Allocation

Technique: Cooperative MARL.

- **Scenario:** A network has 100 nodes but budget for only 5 high-interaction honeypots.
- **Agents:** Blue Agents distributed across the network observe local traffic.
- **Action:** Agents communicate to decide where to dynamically move the honeypots to maximize the probability of trapping an active attacker.
- **Type:** Cooperative Game (Agents share the reward of catching the hacker).

Example Dataset (for Environment creation)

Dataset: Honeypot Attack Logs (Kaggle)
Logs useful for training the traffic generator in the simulation.


AI-DRIVEN CYBERSECURITY & STRATEGIC DEFENSE:
MULTI-AGENT HONEYPOT ALLOCATION
TECHNIQUE: COOPERATIVE MARL
SCENARIO & AGENTS
NETWORK: 100 NODES, BUT ONLY 5 HIGH-INTERACTION HONEYPOTS AVAILABLE
OBSERVING LOCAL TRAFFIC. MUST MAXIMIZE TRAP PROBABILITY
HONEYPOT
ACTION: DYNAMIC COORDINATION
ACTION: AGENTS COMMUNICATE TO DECIDE WHERE TO DYNAMICALLY MOVE THE HONEYPOTS
TYPE: COOPERATIVE GAME
ACTIVE ATTACKER
SHARED REWARD
COOPERATIVE GAME: AGENTS SHARE THE REWARD OF CATCHING THE HACKER
EXAMPLE DATASET FOR ENVIRONMENT CREATION
HONEYPOT ATTACK LOGS (KAGGLE)
LOGS USEFUL FOR TRAINING THE TRAFFIC GENERATOR IN THE SIMULATION

Summary of Techniques & Datasets

Category	Technique	Application	Dataset
Supervised	CNN / XGBoost	Malware Detection	Microsoft Malware Prediction
Supervised	LSTM (RNN)	Traffic Forecasting	CIC-IDS2017
Supervised	NLP / SVM	Phishing/Spam	Enron Email Corpus
Unsupervised	Isolation Forest	Anomaly Detection	NSL-KDD
Unsupervised	Clustering	User Behavior (UEBA)	CERT Insider Threat
Semi-Supervised	GANs	Password Cracking	RockYou.txt
Semi-Supervised	Label Prop.	Malicious URLs	Malicious URLs
RL / MARL	DQN	Auto Pen-Testing	CyberBattleSim (Env)
RL / MARL	Multi-Agent	Honeypot Allocation	Honeypot Logs

Generative AI: The New Cyber Frontier

Definition: Foundation models (like GPT-4, Llama 3) that can generate code, text, and scripts.

Offensive Capabilities

- **Hyper-Personalized Phishing:** Scaling social engineering with perfect grammar and context.
- **Polymorphic Malware:** Rewriting code structure automatically to evade signature detection.

Defensive Capabilities

- **Automated Remediation:** Not just finding bugs, but rewriting code to fix them.
- **Threat Intel Summarization:** Distilling millions of logs/reports into actionable executive summaries.

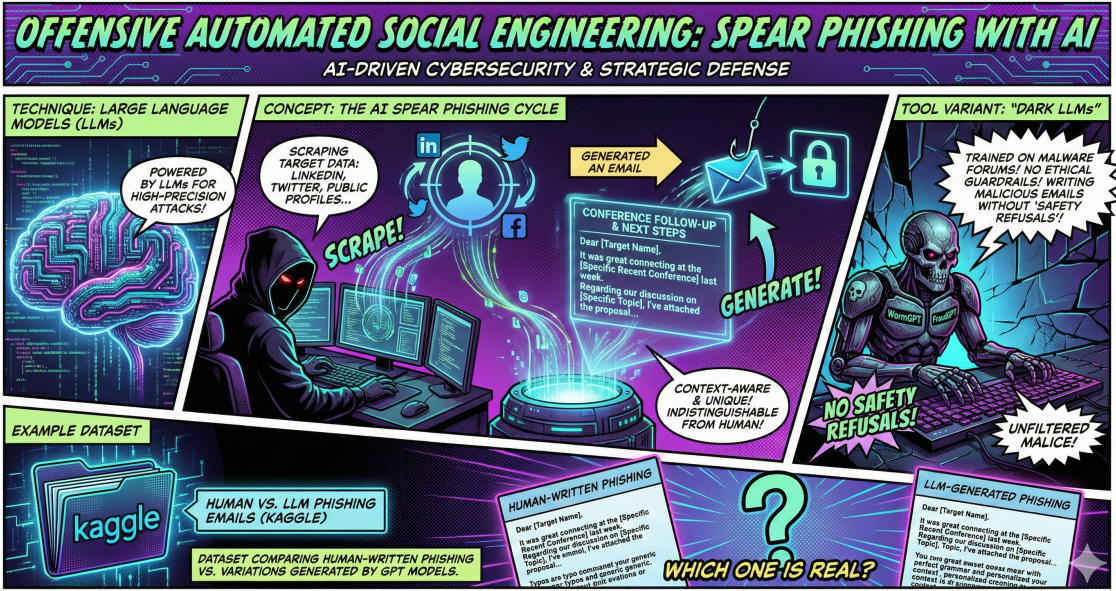
Offensive Ex 1: Automated Social Engineering

Technique: Large Language Models (LLMs) for Spear Phishing.

- **Concept:** Attackers scrape a target's LinkedIn/Twitter to build a profile. An LLM then generates a unique, context-aware email (e.g., referencing a specific recent conference) that is indistinguishable from human correspondence.
- **Tool Variant:** *"Dark LLMs"* (e.g., WormGPT, FraudGPT) are trained on malware forums and lack ethical guardrails, allowing them to write malicious emails without "safety refusals."

Example Dataset

Dataset: Human vs. LLM Phishing Emails (Kaggle)
Dataset comparing human-written phishing vs. variations generated by GPT models.


OFFENSIVE AUTOMATED SOCIAL ENGINEERING: SPEAR PHISHING WITH AI
AI-DRIVEN CYBERSECURITY & STRATEGIC DEFENSE
TECHNIQUE: LARGE LANGUAGE MODELS (LLMs)
POWERED BY LLMs FOR HIGH-PRECISION ATTACKS!
CONCEPT: THE AI SPEAR PHISHING CYCLE
SCRAPING TARGET DATA: LINKEDIN, TWITTER, PUBLIC PROFILES...
SCRAPE!
GENERATED AN EMAIL
CONFERENCE FOLLOW-UP & NEXT STEPS
Dear [Target Name],
It was great connecting at the [Specific Recent Conference] last week.
Regarding our discussion on [Specific Topic], I've attached the proposal...
GENERATE!
CONTEXT-AWARE & UNIQUE/ INDISTINGUISHABLE FROM HUMAN!
TOOL VARIANT: "DARK LLMs"
TRAINED ON MALWARE FORUMS! NO ETHICAL GUARDRAILS! WRITING MALICIOUS EMAILS WITHOUT 'SAFETY REFUSALS'!
WormGPT
FraudGPT
NO SAFETY REFUSALS!
UNFILTERED MALICE!
EXAMPLE DATASET
kaggle
HUMAN VS. LLM PHISHING EMAILS (KAGGLE)
DATASET COMPARING HUMAN-WRITTEN PHISHING VS. VARIATIONS GENERATED BY GPT MODELS.
HUMAN-WRITTEN PHISHING
Dear [Target Name],
It was great connecting at the [Specific Recent Conference] last week.
Regarding our discussion on [Specific Topic], I've attached the proposal...
Typos are typo comsnanet your generic
WHICH ONE IS REAL?
LLM-GENERATED PHISHING
Dear [Target Name],
It was great connecting at the [Specific Recent Conference] last week.
Regarding our discussion on [Specific Topic], Topic, I've attached the proposal...
You ivre great swset poss mear with perfect grammar and personalized your context, personalized your context is ar

Offensive Ex 2: Malware Mutation (Polymorphism)

Technique: Code Generation Models (e.g., CodeLlama fine-tuned).

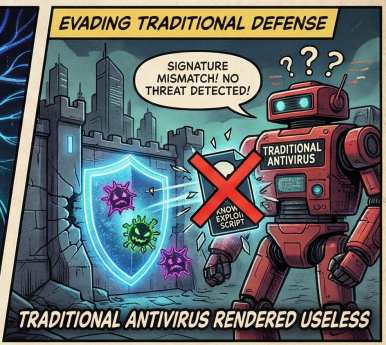
- **Concept:** Attackers take a known exploit script and ask the LLM to "Rewrite this code to achieve the same function but using different libraries/syntax."
- **Result:** The logic remains the same, but the *file hash* and *byte signature* change completely, rendering traditional Antivirus useless.

Benchmark for Evaluation

Dataset: Purple Llama CyberSecEval (Meta)
A benchmark to test if an LLM will comply with requests to generate cyberattack code.

AI-DRIVEN CYBERSECURITY & STRATEGIC DEFENSE: MALWARE MUTATION (POLYMORPHISM)

TECHNIQUE: CODE GENERATION MODELS (e.g., FINE-TUNED CODELLAMA)



Offensive Ex 3: Xillen Stealer v4-v5

Technique: Polymorphism

- Xillen Stealer v4-v5 uses advanced features to evade AI detection, steal credentials, cryptocurrency, and sensitive data across browsers, password managers, and cloud environments.
- With polymorphic engines, container persistence, and behavioral mimicking, this Python-based malware highlights evolving threats and future AI integration in cybercrime campaigns.
- Xillen Stealer is marketed on Telegram, with different licenses available for purchase.

Introduction to Polymorphism

Defining Polymorphism

In the context of malware like **Xillen Stealer**, polymorphism is the ability of code to constantly mutate its identifiable features (its signature) while maintaining its original malicious functionality.

Think of a spy infiltrating a building:

- **The Disguise (Appearance):** The spy wears different clothes, uses a different accent, and enters through different doors every day. To the security guard (Antivirus), they look like a stranger.
- **The Mission (Functionality):** Regardless of the disguise, the objective (stealing data) remains exactly the same.

The Core Concept: Changing the Fingerprint

Traditional antivirus software relies on identifying specific *fingerprints* or hashes (unique strings of bits).

Static Malware

- If you download a standard virus 10 times, the file looks identical 10 times.
- Once the signature is flagged, it is blocked globally.

Polymorphic Malware

- If you download Xillen Stealer 10 times, the **Polymorphic Engine** ensures the file looks different 10 times.
- Binary structure, file size, and keys change.
- **Result:** A completely new file hash for every instance.

Inside the Polymorphic Engine

Xillen Stealer achieves mutation through several technical mechanisms:

- **Encryption & Decryption Loops:**
  - The malicious payload (crypto-stealer) is encrypted.
  - The only *clear* code is a small decryption routine.
  - The engine generates a random key and a slightly different decryption routine for every infection.
- **Junk Code Insertion:** The engine inserts *dead code* or NOPs (No Operation) that do nothing but alter the file structure and offset byte sequences.
- **Instruction Substitution:** Swapping CPU instructions for functional equivalents (e.g., replacing `add 1` with `subtract -1`).

The Danger of Python-Based Polymorphism

Xillen Stealer utilizes Python, which introduces specific advantages for attackers:

1. **Dynamic Obfuscation:**
   - Scrambling variable names.
   - Encoding strings.
   - Restructuring script syntax dynamically before compilation.
2. **Rapid Iteration:** Python is easy to write and modify, allowing attackers to update their polymorphic engines rapidly.
3. **Packaging Complexity:** Python payloads are often "packed" into standalone executables (using tools like PyInstaller). These files are naturally large and messy, which inherently complicates analysis for security tools.

Evading AI Detection

Polymorphism is specifically effective against modern AI defenses.

Bypassing Detection

- **Poisoning the Data:** By constantly shifting features, the malware creates "noise" reducing the confidence of AI classification models.
- **Defeating Static Analysis:** Because the main code is encrypted and the structure is scrambled, the AI sees benign-looking junk code rather than malicious logic.

While polymorphism hides appearance, mimicking hides *intent*. Xillen may:

- Pause execution to simulate human reading time.
- Mimic mouse movements.
- Wait for specific browser activity before striking.

Summary: Standard vs. Polymorphic Malware

Feature	Standard Malware	Polymorphic Malware
File Signature	Constant (Static Hash)	Constantly Changing (Dynamic Hash)
Appearance	Identical on every machine	Unique on every machine
Detection	Easily caught by signature-based AV	Requires Heuristic or Behavioral analysis
Encryption	Often none or static	Dynamic encryption with variable keys

Table: Comparison of Malware Architectures

Offensive Ex 3: Xillen Stealer v4-v5

Technique: AI classification

- The `AITargetDetection` class is intended to use AI to detect high-value targets based on weighted indicators and relevant keywords defined in a dictionary.
- These indicators include “high value targets”, like cryptocurrency wallets, banking data, premium accounts, developer accounts, and business emails.

```python
def enable_ai_detection(self):
    """Enable AI target detection"""
    print("🤖 Enabling AI Target Detection...")

    try:
        self.ai_active = True

        # Initialize AI models
        self._initialize_ai_models()

        # Load training data
        self._load_training_data()

        # Start background analysis
        self._start_background_analysis()

        print("AI Target Detection enabled!")
```

Figure: Xiller Stealer AI Target Detection Class

Offensive Ex 3: Xillen Stealer v4-v5

Technique: statistic noise injection

- `AIEvasionEngine` is a module designed to help malware evade AI-based or behavior-based detection systems, such as EDRs and sandboxes.
- It mimics legitimate user and system behavior, injects statistical noise, randomizes execution patterns, and camouflages resource usage. Its goal is to make the malware appear benign to machine learning detectors.

```python
def create_entropy_variance(self):
    try:
        entropy_levels = [0.3, 0.7, 0.9, 0.5, 0.8]

        for level in entropy_levels:
            data_size = int(1024 * level)

            if level < 0.5:
                data = b'A' * data_size
            else:
                data = os.urandom(data_size)

            _ = hashlib.sha256(data).hexdigest()
            time.sleep(random.uniform(0.01, 0.05))

    except Exception:
        pass
```

Figure: AI evasion function to create entropy variance.

Offensive Ex 3: Xillen Stealer v4-v5

Technique: Noise Injection (`AIEvasionEngine`)

- **Behavioral Mimicking**: Simulates user actions (mouse movement, fake browser use, file,network activity)
- **Noise Injection**: Performs random memory, CPU, file, and network operations to confuse behavioral classifiers
- **Timing Randomization**: Introduces irregular delays and sleep patterns to avoid timing-based anomaly detection
- **Resource Camouflage**: Adjusts CPU and memory usage to imitate normal apps (such as browsers, text editors)
- **API Call Obfuscation**: Random system API calls and pattern changes to hide malicious intent
- **Memory Access Obfuscation**: Alters access patterns and entropy to bypass ML models monitoring memory behavior

Defensive Ex 1: Automated Vulnerability Repair

Technique: LLM-driven Patching (APR - Automated Program Repair).

- **Concept:** Instead of just flagging a vulnerability (SAST), the LLM proposes the actual code fix.
- **Workflow:**

$$\text{Vulnerable Code} + \text{Error Log} \xrightarrow{LLM} \text{Secure Code Candidate}$$

- **Application:** Automatically closing SQL Injection holes by rewriting raw queries into parameterized queries.

Example Dataset

Dataset: LLM Code Vulnerability Detection (Zenodo)
Dataset specifically designed to test LLMs on detecting and fixing CWE vulnerabilities.

Defensive Ex 2: Threat Intelligence Summarization

Technique: NLP Event Extraction & Summarization.

- **Problem:** SOC analysts receive thousands of alerts and lengthy PDF reports daily.
- **Solution:** An LLM reads unstructured reports (blogs, dark web feeds) and extracts structured data (Actor: APT29, Target: Finance, IP: 192.168.x.x) into a JSON format for the firewall.

Example Dataset

Dataset: CASIE (CyberAttack Sensing & Info Extraction)
Annotated dataset for extracting cybersecurity events from text.

Extended Summary: Generative Models

Category	Technique	Application	Dataset / Benchmark
GenAI (Offense)	Dark LLMs	Auto-Phishing	Human vs. LLM Phishing
GenAI (Offense)	Code Gen	Malware Mutation	Purple Llama CyberSecEval
GenAI (Defense)	APR (LLM)	Auto-Patching	LLM Vuln. Detection
GenAI (Defense)	NLP / Summ.	Threat Intel Parsing	CASIE Dataset

Thank You

Questions?