Introduction to Artificial Intelligence Security

Imre Lendák Péter Kiss

March 26, 2026

Contents

1 Introduction **1**
- 1.1 The AI-related OWASP Top 10 lists . . . 2
- 1.2 Representative Actors . . . 3
- 1.3 Attacker types . . . 4
- 1.4 Attack types . . . 4
- 1.5 Attacks against generative AI . . . 5
- 1.6 GAI in cybercrime . . . 6
  - 1.6.1 Deepfakes . . . 6
  - 1.6.2 Attack automation. . . . 6
- 1.7 AI security ontology . . . 6
- 1.8 AI governance . . . 6
  - 1.8.1 Trustworthy AI . . . 8
- 1.9 Book structure . . . 9

2 Data centre security **11**
- 2.1 Physical security . . . 11
  - 2.1.1 External controls . . . 11
  - 2.1.2 Internal controls . . . 12
  - 2.1.3 Five levels of physical security. . . . 12
- 2.2 Personnel security . . . 14
  - 2.2.1 Hiring process . . . 14
  - 2.2.2 Ongoing monitoring during employment. . . . 14
  - 2.2.3 Contract termination . . . 15
- 2.3 Secure operations . . . 15
  - 2.3.1 Data protection . . . 15
  - 2.3.2 Data Privacy . . . 16
  - 2.3.3 Security Monitoring . . . 17
  - 2.3.4 Supply chain security . . . 19
- 2.4 Risk management . . . 20
  - 2.4.1 Identifying systems and endpoints . . . 21
  - 2.4.2 Threats and threat sources . . . 21
  - 2.4.3 Vulnerabilities . . . 22
  - 2.4.4 Likelihood . . . 24
  - 2.4.5 Impact . . . 25

2.4.6 Risk and risk matrix 26

3 Input data manipulation **29**
3.1 Evading spam detection 29
3.2 Evading image classification 30
3.2.1 White-box attacks 30
3.2.2 Black-box attacks 32

4 Confidentiality attacks **37**
4.1 Model inversion 37
4.2 Membership inference 39
4.3 Model theft 40

5 Integrity attacks **43**
5.1 Data poisoning 43
5.1.1 Data poisoning classification systems 45
5.2 Model poisoning 46
5.3 Supply chain attacks against the AI 47

6 Availability attacks **49**
6.1 Model-focused Availability Attacks 49
6.2 Availability Attacks via Upstream Dependencies 49
6.3 Sponge attacks 50

7 Protecting AI systems **53**
7.1 Data protection in AI systems 53
7.1.1 Synthetic data generation 54
7.1.2 Differential privacy 55
7.2 Securing AI models 55
7.2.1 Adversarial training 56
7.2.2 Model regularization 57
7.2.3 Ensemble learning 58
7.2.4 Model watermarks 60
7.2.5 Explainable AI (XAI) 62
7.3 AI system monitoring 63
7.3.1 Training phase monitoring 64
7.3.2 Operations phase monitoring 65
7.3.3 Cut-off thresholds 68

8 Securing generative and agentic AI **73**
8.1 Protecting Generative AI Systems 73
8.1.1 LLM01: Prompt Injection 73
8.1.2 LLM02: Sensitive Information Disclosure 75
8.1.3 LLM03: Supply Chain 76
8.1.4 LLM04: Data and Model Poisoning 76
8.1.5 LLM05: Improper Output Handling 77
8.1.6 LLM06: Excessive Agency 78

8.1.7 LLM07: System Prompt Leakage . . . 79
8.1.8 LLM08: Vector and Embedding Weaknesses . . . 80
8.1.9 LLM09: Misinformation . . . 81
8.1.10 LLM10: Unbounded Consumption . . . 81
8.1.11 Security Controls in Generative and Agentic AI . . . 81
8.2 Generative AI in cybercrime . . . 82
8.2.1 Automated Reconnaisance . . . 82
8.2.2 Enhanced Social Engineering . . . 83
8.2.3 Automated Malware . . . 83

9 AI incident response and forensics 85
9.1 Incident response in AI systems . . . 85
9.1.1 Incident response basics . . . 85
9.1.2 Incidents in AI systems . . . 86
9.1.3 AI incident response process . . . 87
9.2 AI forensics defined . . . 88
9.2.1 The brief history of forensics science . . . 88
9.2.2 The digital forensic process . . . 89
9.2.3 AI forensics goals . . . 90
9.2.4 AI Forensics Goals . . . 91
9.2.5 Evidence . . . 91
9.2.6 Challenges . . . 92

10 Federated learning 95
10.1 FedSGD and FedAvg . . . 95
10.2 Security of FL . . . 96
10.3 Model theft . . . 97
10.4 Poisoning Attacks in FL . . . 97
10.4.1 Model Poisoning Attacks . . . 97
10.4.2 Data Poisoning Attacks . . . 97
10.5 Data Privacy Attacks . . . 101
10.5.1 Data reconstruction from gradients . . . 102
10.5.2 Linear layer leakage attacks . . . 103
10.5.3 Defense techniques . . . 103
10.6 Cross-Silo FL Security . . . 105
10.6.1 Comparison of Security and Privacy Threats . . . 105

Appendices 110

A Federated Learning Threat Model and Notation 113
A.1 Federated learning setup (round-based training) . . . 113
A.2 Adversary roles and capabilities . . . 113
A.2.1 Malicious clients (Byzantine participants) . . . 114
A.2.2 Sybil attacker (many fake clients) . . . 114
A.2.3 Curious or malicious server . . . 114
A.3 Observation regimes (what the adversary sees) . . . 114

A.4 Security goals in FL 114
A.5 Defense families (where defenses operate) 115

B Federated Learning Poisoning Attacks 117
B.1 Taxonomy of poisoning in federated learning 117
B.2 Attack/defense pairing I: Data poisoning in FL 118
B.2.1 Attacks: dirty-label and label-flipping/backdoor-style poisoning 118
B.2.2 Defenses: robust losses and data-based anomaly detection (often privacy-invasive) 118
B.3 Attack/defense pairing II: Model poisoning in FL 119
B.3.1 Attacker goals and targeted objectives 119
B.3.2 Stealth: why poisoning must evade server-side checks . . . 119
B.3.3 Attack: optimization-based model poisoning with explicit boosting 120
B.3.4 Defense: Byzantine-resilient aggregation (mostly for untargeted poisoning) 121
B.3.5 Attack: fake clients and consistent steering (MPAF) . . . 121
B.3.6 Attack: backdoors and model replacement 122
B.4 Verification-based defenses (stronger assumptions) 122

C Gradient/Update Leakage Attacks 123
C.1 OR I: Single-round updates 123
C.1.1 Single data-point (analytical leakage for simple architectures) 124
C.1.2 Deep Leakage from Gradients (DLG) and iDLG 124
C.1.3 Arbitrary architecture for single datum 125
C.1.4 Reconstruction of a batch of data (single client batch) . . 126
C.1.5 Minimizing cosine distance (direction matching) 126
C.1.6 Defenses for regime I 127
C.2 OR II: Aggregate-only updates 127
C.3 OR III: Malicious protocol design 128
C.3.1 Linear-layer leakage attacks (analytical reconstruction perspective) 128
C.3.2 Trap Weights 128
C.3.3 Robbing the Fed 129
C.3.4 Loki 129
C.3.5 Defenses for regime III 129

D Inference on private data 131

Chapter 1

Introduction

The goal of this textbook is to provide readers with diverse backgrounds (engineering, computer science and other) with relevant knowledge about the cyber security and the wider trustworthiness implications of applied artificial intelligence (AI). Trustworthiness itself is a complex term and incorporates at least privacy, safety, reliability, validity, interpretability and explainability.

Artificial intelligence (AI) refers to a solution in which an information system, or more narrowly, a computer, makes decisions based on the information it collects. Machine learning is a narrower term, and generally refers to systems, and primarily algorithms and models, in which one or more computers form a model suitable for decision-making based on some training data set.

Many people view artificial intelligence and machine learning as novel concepts, and some equate them with generative artificial intelligence (e.g., ChatGPT, Gemini, Dall-E), but these concepts are traditional research areas in computer science and have a long tradition. Fuzzy logic has been with us since 1965 (Zadeh, 1965), while the history of artificial neural networks (on which generative AI is also based) dates back at least to the 1940s-1950s, and the well-known error backpropagation procedure used during training was born in 1970 (Linnainmaa, 1970).

Although artificial intelligence security (AI) is a relatively novel domain, we saw a large but fragmented scientific interest in the field since at least 2014 when researchers at Google published their findings on relatively easy-to-do evasion attacks. Multiple valuable resources appeared in AI security in the 2020s, namely the OWASP Top 10 Machine Learning Security OWASP (2023), the NIST report on AI security NIST (2024) or the OWASP AI Exchange[1].

We will loosely follow the methodology implemented by the authors of the NIST report and structure our content into a data center security, adversarial data, attacks against elements of the CIA triad, security measures protecting the AI systems and federated learning security.

Within this introductory chapter we briefly present the security challenges

[1] `https://owaspai.org/`

that can be identified in the field of artificial intelligence, mainly focusing on the types of attackers and attacks.

1.1 The AI-related OWASP Top 10 lists

The Open Web Application Security Project (OWASP) expert organization is probably best known for its top ten lists of most important vulnerabilities in different types of systems. They published the Machine Learning Security Top Ten in 2023 [2], which we overview in table 1.1.

Table 1.1: OWASP Top 10: Machine Learning Security (2023)

ID	Title	Brief description
ML01	Input Manipulation Attack	Attackers manipulate the AI system inputs to achieve their goals.
ML02	Data Poisoning Attack	Integrity breach of training data critical in most forms of AI.
ML03	Model Inversion Attack	Usually training data inversion based on analysing an AI model or its responses.
ML04	Membership Inference Attack	The attackers analyse the AI model or its responses and find out if a certain observation was included in the training set.
ML05	Model Theft	The attackers steal the model (binary).
ML06	AI Supply Chain Attacks	The attackers breach the supply chain of the applied AI system and (usually) breach the integrity of key artifacts (data, models, libraries).
ML07	Transfer Learning Attack	The attacker trains a model on one task and then fine-tunes it on another task.
ML08	Model Skewing	The attacker modifies the statistical characteristics of the training dataset.
ML09	Output Integrity Attack	The attackers obtain access to the AI system and maliciously alter its outputs.
ML10	Model Poisoning	The attackers breach the integrity of the applied AI model itself.

The high interest towards large language models (LLM) prompoted OWASP to publish their LLM-specific top 10 list as well[3]. It is somewhat similar to the more general ML security list, but also contains LLM-specific vulnerabilities and tailored mitigations.

[2] `https://owasp.org/www-project-machine-learning-security-top-10/`

[3] https://genai.owasp.org/resource/owasp-top-10-for-llm-applications-2025/

Table 1.2: OWASP Top 10 for LLM Applications 2025

ID	Title	Brief description
LLM01	Prompt Injection	User prompts alter the intended operation of an LLM.
LLM02	Sensitive Information Disclosure	LLMs expose sensitive data (personally identifiable information, code, algorithms) in their outputs .
LLM03	Supply Chain	Vulnerabilities introduced into LLMs via training data, third-party models, and/or deployment platforms.
LLM04	Data and Model Poisoning	Vulnerabilities are introduced into LLMs via altered training, pre-training or fine-tuning data, as well as third-party models.
LLM05	Improper Output Handling	Inadequate validation and sanitization of LLM outputs.
LLM06	Excessive Agency	The agency granted to an LLM (e.g., accessing external systems) leads to unintended outcomes.
LLM07	System Prompt Leakage	System prompts used to manage the LLM's operation leak sensitive information.
LLM08	Vector and Embedding Weaknesses	Vulnerabilities in vector and embedding (i.e., machine comprehension of text) generation, storage, or retrieval.
LLM09	Misinformation	The LLM produces incorrect or misleading information which appears credible and can lead to reputational damage and legal liability.
LLM10	Unbounded Consumption	Excessive use of an LLM leads to resource exploitation, denial of service and/or high costs in cloud computing environments.

1.2 Representative Actors

Although at first glance modern generative AI systems seem as yet another web-based, user-facing solutions offered to end users by large corporations, in truth there are many different actors involved in the planning, development, deployment, maintenance and use of AI systems. We define some of the key actors loosely based on the NIST AI Risk Management Framework (AI, 2023).

We first need to differentiate the organizations that develop AI systems from scratch from those that only integrate them from existing components. The first employ AI researchers, data scientists and data engineers in technical roles

geared towards building the models. But a model itself is not yet a true source of revenue, it needs to be wrapped in a graphical user interface or application programming interface, which are built by software engineers, tested by quality assurance experts and DevOps engineers bridging the gap between development and operations. All of them are managed by product managers responsible for the development of the AI system's components, as well as project managers who ensure that customer-facing AI systems are developed and integrated efficiently. These roles can be relevant in both AI model builders and integrators. AI system integrators largely rely on 3rd-party suppliers of components and in large projects might hire external organizations as system integrators.

When building AI models trained on data about humans or systems which will be directly interfacing humans, then additional, less technical actors come into play. Socio-cultural and human factor experts look beyond the technical features of AI systems and consider how they impacts societies, communities or workplaces where they are deployed. Legal and governance experts are familiar with the legislative and regulatory ecosystem in which an AI system will be deployed. Additionally, they are called in when there as incident which leads to penalties or legal action.

On a wider scale, policymakers might be involved in or impacted by AI systems when they develop rules and regulations. Standards organizations are involved in the development of standards and guidelines which ensure that the elements of AI systems built by different solution providers are able to interact through standardized interfaces. Trade associations allow smaller organizations to pool their bargaining weights and obtain preferable or at least fair treatment when dealing with large corporations selling AI services. On the other hand, if an AI system is involved in crime, then law enforcement agents specializing in cybercrime become active actors as well. As AI systems might impact the environment and different communities, we might have to deal with non-government organizations or other bodies formed by these types of actors.

1.3 Attacker types

The relevant scientific and professional literature usually identifies three types of attackers targeting applied AI systems: beginners (also known as amateurs or script kiddies), advanced and sophisticated cyber adversaries. They mainly differ in their knowledge of AI systems, the availability of compute resources necessary to implement an attack, as well as the level of access they have to the target AI system in operations (or less frequently in training).

1.4 Attack types

There are three types of attacks based on level of access to the target AI system. The most common are the so-called black box attacks, in which the attackers obtain access to an operational AI model either via a web interface or an ap-

plication programming interface (API). They do not possess information about the tuple (training/validation/testing data, algorithm, hyperparameters, model architecture and parameters).

In white-box attacks, the attackers obtain full access to the elements of the above tuple. This access can be gained malevolently or benevolently. Malicious, unwanted access can be gained either by hacking into the information system hosting the AI model or with the help of malicious insiders. Law-abiding white-box access can be obtained to model and data if they are open-source. Similarly, algorithm and hyperparameter values can also be made public by model operators.

Grey-box attacks are somewhere between white and black boxes. In such scenarios the attackers obtain limited access to the elements of the above tuple. For example, the model might be open-source or leaked by a malicious insider, but all other elements of the tuple might remain opaque.

1.5 Attacks against generative AI

In the context of ChatGPT, Gemini, Dall-E and similar systems, we almost always assume a black-box attack and it must be taken into account that these are much more complex systems than spam classification or image recognition models.

In the training phase, generative AI developers typically use large amounts of data collected on the open Internet. If the attackers know the sources of the data used in the training phase, they can insert manipulated data into them.

Direct prompt injection attacks are typical of the model exploitation phase, the aim of which is to send inputs or input sequences to the generative AI system, which cause the system to perform an action not planned by the developers. In the early stages of generative AI, there were known attacks in which attackers forced the system to give harmful or even illegal responses by sending a few prompts. In LLM jailbreaking the purpose of prompt injection is to change the so-called persona of the generative AI system, in which the system can switch to a role that is otherwise unavailable to users during a given interaction, for example, changing from a cybersecurity researcher role to a cyberterrorist role; or from a regular to user to a LLM superuser/administrator. After a successful jailbreak the LLM provides responses accordingly to its current context, which might lead to sensitive data exposure, resource exhaustion or other unintended outcomes.

Since generative AI systems are already known for their significant energy consumption, both in the training and use phases, it is important to analyze the possibility of denial of service (AI system overload) attacks and prepare these systems to effectively identify and deal with them. Otherwise, system operators can expect higher than usual electricity usage and bills after overload attacks.

1.6 GAI in cybercrime

1.6.1 Deepfakes

Deepfake is the use of generative AI to generate fake images, audio, or video material. Deepfake poses a serious challenge in cybersecurity because its use allows cybercriminals to mislead human resources in various ways and even circumvent identification systems. Deepfake is also used for political purposes, such as creating fake videos of high-ranking politicians. Identifying and filtering deepfake material is a separate research subfield in AI security (Rana et al, 2022).

1.6.2 Attack automation.

Cybercrime is a multibillion business. One of the key obstacles in its further growth is the lack of human resources. This comes as no surprise as the ISC2 yearly reports continue to show that millions of information security professionals are missing on the job market and this large work-force gap must impact the operations of cybercriminal groups as well, which practically means that they must lack human resources as well. Under these contstraints, cybercriminals can largely benefit from using AI (mainly generative AI) to automate data collection about potential targets (recon or reconnaisance phase), generate convincing text messages used in phishing campaigns, automate vulnerability search and the generation of scripts, malware and other source-code used during attacks (weaponization phase).

1.7 AI security ontology

A simplified ontology of the security of applied AI systems is shown in Figure 1.1. We already discussed attack and attacker types, which are key mid-tier elements in our ontology as well. Apart from them, the figure also depicts attack/attacker strategy, possible targets, visibility, as well as the distinction of attacks occuring in physical or cyber space.

1.8 AI governance

Applied artificial intelligence is not a new concept, but it only really entered the mainstream with the availability of powerful generative AI systems. The publication of ChatGPT in late 2022 [4] was a turning point after which experts, government stakeholders and the general public started to use these systems widely and discuss the possible real-life implications of their use. This is not

[4] `https://www.forbes.com/sites/bernardmarr/2023/05/19/a-short-history-of-chatgpt-how-we-got-to-where-we-are-today/`

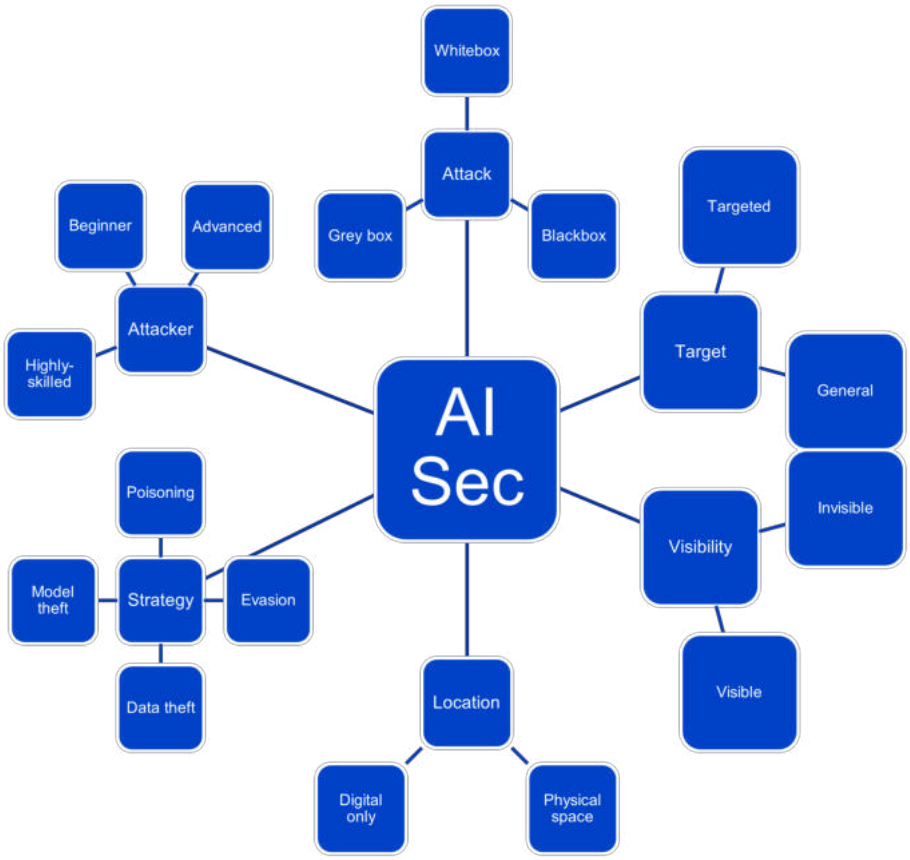


Figure 1.1: Simplified AI security ontology

surprising, considering that the admitted ultimate goal of OpenAI (the developer of ChatGPT) and other companies and organizations is to develop artificial general intelligence (AGI), capable to solve a wide range of different problems.

The European Union leads in the field of AI regulation with the AI Act European Parliament (2023). It is the first legal framework, that ensures that AI systems used in the European Union are safe, transparent, and ethical. It entered force in August 2024 and it follows a risk-based approach and classifies AI into four categories: Unacceptable Risk, High Risk, Limited Risk, and Minimal Risk. Unacceptable Risk systems, such as government-run social scoring or real-time facial recognition in public spaces, are strictly banned since February 2025. High-Risk systems (e.g. AI in healthcare, hiring, or critical infrastructure) must meet rigorous standards for data quality, human oversight, and cybersecurity. Limited Risk AI, such as chatbots are subject to transparency obligations and must clearly inform users that they are interacting with a machine. Minimal Risk systems do not have to meet the above-listed requirements.

1.8.1 Trustworthy AI

The European Commission set up a high-level expert group on artificial intelligence as early as 2018 wit the task to define the key requirements which trustworty AI systems must meet (on Artificial Intelligence, 2019). The expert group defined the following key requirements:

1. **Human agency and oversight.** AI systems need to implement proper human oversight mechanisms with human-in-the-loop (HIL), human-on-the-loop (HOL), and human-in-command approaches.

2. **Technical robustness and safety.** AI systems need to be resilient and secure, thereby ensuring that any unintentional harm they cause is minimized or prevented entirely.

3. **Privacy and data governance.** AI systems implement proper privacy and governance mechanisms, which ensure data quality and integrity of the data.

4. **Transparency.** Data used to test/train/validate, system and AI business models should be transparent. AI system decisions should be explained in a manner adapted to the stakeholder concerned. Humans need to be aware that they interact with an AI system, as well as informed of its capabilities and limitations.

5. **Diversity, non-discrimination and fairness.** AI system developers and operators ensure that unfair bias is avoided.

6. **Societal and environmental well-being.** AI systems are sustainable and environmentally friendly, thereby ensuring that they benefit all human beings, including future generations. Their social and societal impact is carefully considered.

7. **Accountability.** AI system owners and operators are accountable for their AI systems and their outcomes. Auditable AI systems can be inspected by internal or external experts, who are able to verify and assess an AI system's behavior, data, and logic.

1.9 Book structure

The rest of this book is structured into a chapter on data centre security, followed by chapters closely focusing on different key challenges in AI security, namely input data manipulation, as well as confidentiality, integrity and availability attacks targeting systems employing some form of AI. The book also includes chapters focusing on security controls specific to AI systems, AI forensics and federated learning (FL) security.

Bibliography

AI, N. (2023). Artificial intelligence risk management framework (ai rmf 1.0). *URL: https://nvlpubs. nist. gov/nistpubs/ai/nist. ai*, pages 100–1.

European Parliament (2023). EU AI Act: First regulation on artificial intelligence.

Linnainmaa, S. (1970). The representation of the cumulative rounding error of an algorithm as a taylor expansion of the local rounding errors.

NIST (2024). Adversarial Machine Learning: A Taxonomy and Terminology of Attacks and Mitigations.

on Artificial Intelligence, E. C. H. L. E. G. (2019). Ethics guidelines for trustworthy ai.

OWASP (2023). OWASP Machine Learning Security Top Ten.

Zadeh, L. A. (1965). Fuzzy sets. *Information and control*, 8(3):338–353.

Chapter 2

Data centre security

Although most larger scale data centres seem like impenetrable concrete blocks, their information security remains a continuous challenge to owners and operators. These facilities and their security are critical in the era of cloud computing and large-scale deployments of different artificial intelligence solutions. In this chapter we discuss physical and personnel security, as well as risk-aware secure operations.

2.1 Physical security

Maintaining strict physical security in data centres allows system owners and operators to control people flow outside and inside facilities, as well as to detect and respond to adverse natural events (flood, fire, hurricane) (Tiszolczi, 2019).

We differentiate external and internal physical security controls (aka measures). External controls protect the building up to its walls and entrances, while internal controls limit people and resource flows within the facilities.

The deployment of proper physical security controls increases the probability of maintaining the desired level of data centre uptime measured in tiers 1 to 4. Tier 4 data centres are required to reach 99.995% availability (26.3 minutes of downtime annually) (Uptime Institute, 2025).

2.1.1 External controls

Data centres need to be built in properly chosen geographic locations with low likelihood of adverse natural events (e.g., flood, tornado, hurricane, fire). The space around the facility must be cleared to thereby increase visibility around the location. Appropriate walls, fences and external lighting needs to be installed. Gates, doors, windows, rooftops and basements need to be considered as entrances and properly secured. Guards can be deployed to patrol the immediate facility area, as well as video surveillance (Fennelly, 2013). A somewhat

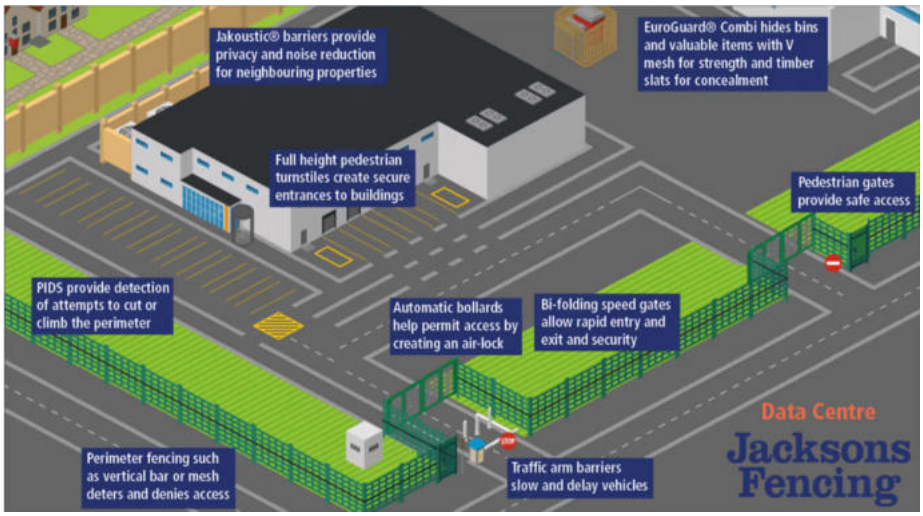


Figure 2.1: External physical security controls. Note: Image adapted from the website Jackson Security)

simplified illustration of external physical security controls put into action protecting a data center are shown in Figure 2.1.

It must be noted that sometimes even the strictest external physical controls are not sufficient to protect facilities from determined and armed attackers (Wikipedia, 2013) or natural disasters.

2.1.2 Internal controls

The primary goals of internal physical security measures are the oversight and control of people flow, detection of intruders and adverse events e.g., fire or flooding. Internal doors can be controlled via multifactor authentication (key cards, biometric sensors). IT and electric infrastructure elements need to be hidden and properly secured. Physical port security is necessary e.g., physical protection of USB ports in servers and operator workstations. Internal video surveillance is an option as well.

2.1.3 Five levels of physical security.

Physical security of data centres and similar facilities (e.g., warehouses, factories, government buildings, embassies, military facilities) can be graded in five levels as shown in Table 2.1.

Note that the security controls are additive and the higher physical security levels incorporate or further improve controls from the lower security levels. The difference in perceived physical security and cost can be very high.

Table 2.1: The five levels of physical security

Security level	Security controls (additive)	Examples
Zero	None	high seas; forest; open fields
Minimum	Basic security controls to keep intruders out: cleared space, lighting, fences, walls.	private residences; urban spaces
Low	Security gate, reinforced locks, bars on windows, alarm.	small shops; storage facilities
Medium	Protection against internal and external threats. Internal surveillance monitors employees, customers and guests. High fence, unarmed security guard, loud alarm.	factories; large retail stores; warehouses;
High	24/7 closed-circuit television (CCTV), perimeter alarm system, gates, co-trolled access in and out, security lighting, armed guards, security dogs, audit, cooperation with law enforcement.	prison; defense; electronics; pharmaceuticals;
Maximum	24/7 security personnel (combat-trained, well-equipped), stricter internal control of movement, tamperproof alarm.	military installations; embassies; nuclear; some gov facilities;

Figure 2.2: Key activities in maintaining skilled and loyal human resources

2.2 Personnel security

Personnel security consists of different measures before and during the hiring process, during employment, and in the sensitive transition phase when a former employee leaves the organisation (see figure 2.2 for an overview of key activities).

2.2.1 Hiring process

It is important to make sure that the candidate hired to work in a secure facility is properly vetted during the hiring process. All claims from the professional biographies of candidates need to be verified via third parties. This might involve direct phone calls to the representatives of former employers, as well as checking candidates via the professional networks of the professionals conducting interviews with future employees. Facts from the curriculum vitae (CV) which need to be checked are at least degrees and certificates verifiable via the issuing institutions, claimed knowledge of spoken languages checked via involving speakers of those languages in interview committees, technical skills can be assessed by involving internal or external experts in the interview process, as well as checking the criminal record of candidates.

2.2.2 Ongoing monitoring during employment.

Define persons to whom employees need to report security incidents. Investigate each failure to report or communicate about suspicious activity withing

secure facilities. Organize periodic security trainings for employees. Send out short security bulletins or other forms of brief summaries about the latest information security challenges representative to your sector. Consider awarding special prizes to employees who strictly follow security policies and procedures. Implement increased monitoring of employee behaviour during sensitive periods e.g., re-assignment, decreased salary or during periods when the organization is laying people off.

2.2.3 Contract termination

When employees leave an organisation for retirement or transfer, they are increased risks that they will either be less adamant in the implementation of security policies, or they might even intentionally cause incidents. It is important to deny access to all compute and communication infrastructure, deactivate or delete all user profiles with a special care provided to any remote access capabilities (e.g., VPN), return company-owned devices (laptop, mobile), as well as keycards, keys and other artifacts allowing physical access to facilities. If there is an increased risk of incident, then a security guard can escort the ex employee(s) outside the facility. If the former employee possessed privileged access rights, then increased security monitoring of accessible devices should be considered.

2.3 Secure operations

2.3.1 Data protection

The data inventory usually consists of physical documents of varying confidentiality levels (public, internal, top secret), electronic documents, electronic databases (schema + data), system configuration (JSON, XML, etc.), and binary files (exe, dll, so, etc.). In environments with applied artificial intelligence, we might additionally have machine learning models, training data, algorithm implementation and hyperparameters.

Data at rest

The key security controls used to protect data at rest are encryption and strict access control. As it is common to store large volumes of data, it is necessary to use efficient algorithms e.g., Advanced Encryption Standard (AES) and similar. Strict access control often relies on multi-factor authentication and zero trust policies. Endpoint protection solutions (anti-malware, host-based intrusion detection) ensure that malware is detected in a timely manner and data integrity and availability is maintained.

Data in transit

The confidentiality of data passing through communication networks can be guaranteed by encrypting all communication channels with unique session keys issued to authenticated and authorised entities. Digital certificates are necessary to ensure that only trusted parties participate in communication. X.509 is a widely used digital certificate standard. The Public Key Infrastructure (PKI) ensures that digital certificates can be assigned to and verified on the open Internet. Both wired and wireless networks utilise AES for encrypting data in transit.

Data in use

The concept of data in use in security is relatively new and entered the mainstream with the advent of cloud computing in which multiple parties of diverse geographic origin can share the same compute and storage infrastructure. Data in use protection solutions ensure that the data does not leak while it is stored in the RAM or the CPU of servers and other devices. Operating systems (OS) strictly separate the memory spaces of processes and randomise the location of both OS and process memory segments. Modern compute hardware used in data centers possess additional, hardware-enabled privacy features like Trusted Execution Environments (TEE) for both CPUs and the latest GPUs[1].

2.3.2 Data Privacy

Many years passed since the entering into force of the first data privacy regulations, among which the most visible was the 1974 Privacy Act in the USA (United States of America (USA), 1974). Modern privacy regulations like General Data Protection Regulation (GDPR) of the European Union (EU) (European Parliament and European Commission, 2016) define the subjects who own sensitive data, data controllers who collect and store data, as well as data processors who process data. A list of example data privacy controls which must be implemented by these actors are listed below:

- **Data minimization.** Data collection must be minimal and aligned with the stated goals of the collector/processor.
- **Data owner involvement.** The data owner must be able to check the data collected, ask for corrections or even deletion.
- **Access control and audit.** The controller/processor strictly control and audit all access to personal data.
- **Legal liability.** The legal representatives (e.g., Chief Executive Officer - CEO or Chief Information Security Officer - CISO) of the data controller and/or processor are legally liable to protect the personal data put into their hands.

[1] `https://www.nvidia.com/en-us/data-center/solutions/confidential-computing/`

According to the GDPR, in case there is a data breach and the investigation shows lack of vigilance on the data controller/processor side, they are liable to pay significant fines when the breach involves personal data of EU citizens. Personal data includes name, address and nationality, as well as any other data which can be linked directly or indirectly to a living individual. Sensitive personal data include genetic, biometric and health data, as well as personal data revealing racial and ethnic origin, political opinions, religious or ideological convictions or trade union membership.

Personally identifiable information (PII) is an often used term in this context and it refers to information based on which an individual can be identified e.g., address and name.

Data privacy is highly relevant in the context of AI security, as the training data often contains personal data. This is important, because we will show that training data can be extracted from artificial intelligence models under certain conditions, thereby leading to novel forms of personal data breaches.

2.3.3 Security Monitoring

Highly skilled attackers could operate undetected in compromised information systems for months after a successful attack and obtaining remote access up until the early 2020s and in most critical infrastructure sectors (energy, transportation, manufacturing and others). This time allowed attackers to move laterally and extend their attack beyond the primary affected hosts and/or networks. In certain cases this even allowed them to cause damage in the physical space during the time they maintained remote access to the compromised system(s). A widely publicized example of this was the cyberattack on the electricity distribution system operators (DSOs) in Kiev (Ukraine) in December 2015, in which the attackers switched off entire power lines via the Supervisory Control and Data Acquisition (SCADA), resulting in power outages at over 100,000 customers (SANS, 2016). An important feature of the attack was that the employees of several affected companies opened malicious email attachments thereby initiating the first steps of the attacks sometime in the summer of 2015, which was roughly six months prior to the actual cyber attacks. Unfortunately, the attackers were not identified during the several months of operations because the attacked electricity providers did not have adequate monitoring systems. The situation was similar in Europe and North America at the time, meaning that a similar attack could have been successful in those geographical regions as well.

Based on the above example, it comes as no surprise that monitoring systems greatly assist in the early identification of cyberattacks similar to those described above as well as in the analysis of cybercriminals' activities. These systems collect data on significant events in a central location, for example, the successful or unsuccessful login of the system administrator to a computer, the installation of patches, the creation of backups (or failure to do so), the identification of malicious software or email content, and the many others, depending on sector, regulatory environment and organization-specific requirements.

Security Operations Center (SOC).

Security Operations Centers (SOCs) are designed to provide 24/7 security monitoring of large critical infrastructures. SOC employees are familiar with the monitored system and can use monitoring data to respond to detected undesirable events, such as new malicious content in emails, suspicious administrator logins, or changes to binary files on specific computers. In large organizations, the Computer Security Incident Response Team (CSIRT) may also operate within or in conjunction with the SOC.

Figure 2.3: Security Operations Center

Security Information and Event Management (SIEM).

Security Information and Event Management (SIEM) systems enable the collection, analysis, visualization of critical events, as well as reporting, and notifications based on those data. SIEMs collect data using so-called probes (or collectors/connectors) installed on the monitored devices. On desktop and server computers, they might collect information from the operating system, anti-malware software, and other endpoint protection devices. In server rooms, they can collect data on physical access, environmental factors (e.g., temperature, humidity), as well as network infrastructure, e.g. switches, routers, IDS

devices. SIEMs store the collected data for a given period of time. In some sectors, sectoral regulatory bodies (usually government agencies) regulate the expected minimum event retention period - for example in the energy sector, this is usually around a year. In addition to storage, SIEM also performs statistical and machine learning-based analysis of data and anomaly analysis, and display data and notifications in a graphical environment for operators.

2.3.4 Supply chain security

The supply chains of different data centres and their tenants can vary widely. Electricity and other utilities need to be available with high reliability, usually via redundant links to utility companies. Contractors working on the building(s) or the network and compute infrastructure within the centres need to be certified for the work carried out and pass additional, rigorous security checks. Hardware components can be bought from trusted partners with strict, tamper-proof production processes ensuring that no hardware Trojans are inserted. Similarly, all software componentes (e.g., operating systems, virtualisation platforms, application software, libraries) needs to be digitally signed, downloaded from trusted sources, scanned and verified in testing or staging environments before deployment.

Data and artificial intelligence-specific supply chains include additional elements. The data used in the model-building (training) process must come from trusted sources and pass well-defined quality control processes. Similarly, pre-trained models and necessary libraries must originate from trusted partners verifiable via digital signatures.

Although there were numerous supply chain attacks, one of the most publicised was the Solarwinds attack in 2020, during which government agencies in the USA deployed system monitoring sofware which included a backdoor inserted by a foreign actor after hacking into the systems of the producer (Wikipedia, 2020). The steps of this (and other similar attacks) are outlined in Figure 2.4.

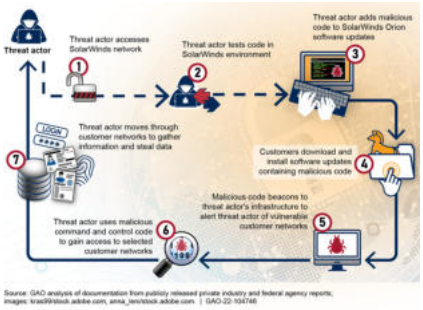


Figure 2.4: 2020 United States federal government supply chain attack

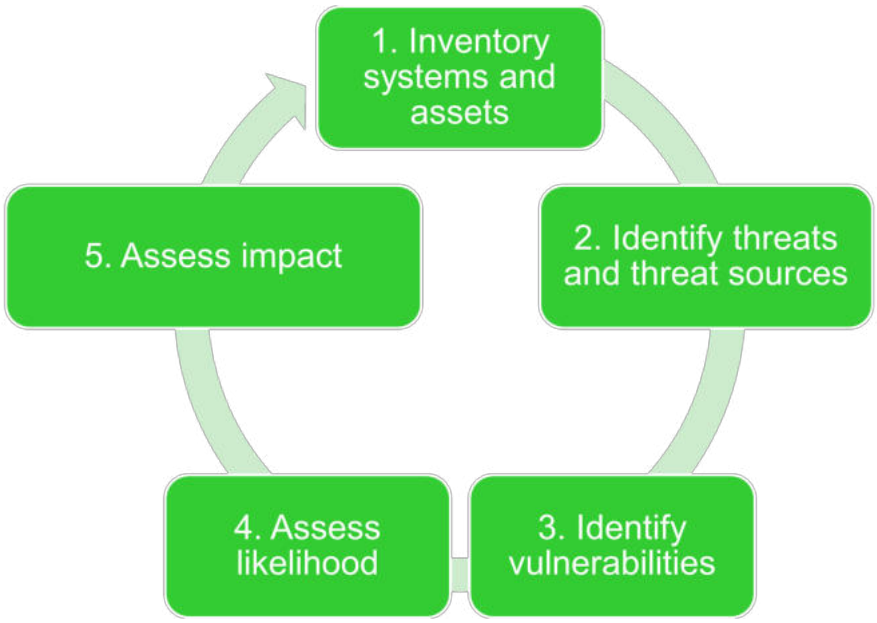


Figure 2.5: The circular risk management process

2.4 Risk management

The stated goal of the NIS2 Directive of the European Union is to protect critical infrastructures in the EU from cyber attacks (European Parliament and European Commission, 2022). It mandates all EU Member States to pass laws and bylaws strictly defining and controlling the development and application of risk-aware security controls in all critical infrastructure sectors (e.g., finance, water, agriculture, telecommunications, transport, energy, space and others).

The risk management process consists of the identification of the following: (1) key systems and their endpoints within the organisation which are critical to the core business or operations, (2) threats and threat sources which might adversely impact normal operation, (3) security vulnerabilities in information and other systems which might allow threat sources initial access to the system(s), (4) likelihood and (5) potential impact of all identified incidents Bodungen et al. (2016). This process is implemented iteratively until a desired risk-aware information security posture is reached (as depicted in figure 2.5.

Insurance companies have decades or even centuries of statistical data on natural disasters, car accidents and life insurance events. Based on this, they can estimate the likelihood of unwanted events and their impact quite accurately. In practice, this means that the probability of an accident affecting a given vehicle and the extent of material damage can be accurately estimated. Unfortunately, such statistics are (usually) not (yet) available in the field of cybersecurity, and therefore the risk assessment methods used in this area are approximate and sig-

nificantly sector-dependent: the information systems of financial systems and banks are constantly attacked by criminals, while wastewater or waste management are generally less attractive targets. Risk assessment requires knowledge of the system under investigation, sector-specific (or industry-specic) knowledge, and appropriate risk analysis methods. Knowledge of industry trends and the system under investigation allows for the identification and description of theoretically feasible attacks.

2.4.1 Identifying systems and endpoints

The first phase of risk analysis is the identification and inventory of systems, assets and endpoints accessible in physical and/or electronic (cyber) space. The level of awareness of assets potentially affected by cybersecurity or other (e.g., failure) incidents tends to vary across industries, with government, military and financial institutions likely to have very detailed inventories, while fast-growing startups, or organizations in sectors like agriculture or manufacturing may have much less complete asset lists.

Physical, electronic, and hybrid endpoints are particularly important as they are the means through which the system can be accessed and through which attackers (both in the physical and cyber space) can gain unauthorized access. All identified endpoints in the system should be listed and described in adequate detail.

Physical endpoint descriptions typically include the following characteristics: identifier, description, geographic location (GPS), and level of security that provides access. In the case of electronic endpoints, in addition to the identifier, a unique MAC and IP address can be specified, as well as a description and type of the device, OS/firmware type, installed and enabled software, and upstream and downstream dependencies. For both physical and electronic endpoints, the expected maximum allowable recovery time in the event of a failure should be specified. This time is mandatory for assets that are particularly important for operations, e.g., transformers in electrical power systems or physical servers in cloud data centres.

2.4.2 Threats and threat sources

The threats which might result in some form of harm in data centres and similar facilities can be grouped in the below-listed categories:

1. Environmental: Fire, flood, tornado, hurricane, corrosion, electromagnetic interference or flare.

2. Physical security: Different forms of spying (listening devices, unauthorized video monitoring), physical breaking and entering, device theft or loss, sabotage or physical (e.g., armed) attack.

3. Personnel security: Loss of critical employees (churn, pension, death),

social engineering, extortion and ransom, corruption, identity theft, repudiation of actions.

4. Legal: Breach of laws, regulations or contracts.
5. Infrastructure providers: Loss of internet access or electricity, supply chain incidents, other critical services (e.g., input of materials in critical manufacturing sectors).
6. Data-related: Data leak, loss or manipulation. Accidental or malicious use of non-trustworthy information.
7. Hardware and software: Destruction of hardware or deletion of software, fake software or hardware (potentially with backdoors), mistakes during HW or SW use, authorisation failures.
8. Infrastructure: Unauthorised access and integrity breaches manipulated network flows, denial of service, malware.

The types of usual threat sources are listed in Table 2.2. Nation states and malicious insiders are usually the most likely to succeed in an attack. Nation-state actors possess both the necessary technical skills and adequate funding to find and exploit vulnerabilities even remotely. Malicious insiders, on the other hand, usually have physical access to the protected zones and can plant unauthorized devices or leak critical credential information, significantly decreasing the complexity of attacks. Nevertheless, most attacks still materialise due to careless or poorly trained employees who fall prey to spear phishing or more sophisticated social engineering or even extortion.

2.4.3 Vulnerabilities

Vulnerabilities can be identified in people, processes or the technologies (PPT) used. Information about vulnerabilities in hardware, firmware, operating system, or software can be found in freely available sources on the Internet. Perhaps the most significant of these sources are the following:

- National Vulnerability Database (NVD), https://nvd.nist.gov/
- Common Vulnerabilities and Exposures (CVE), https://www.cve.org
- Common Vulnerability Scoring System (CVSS), https://nvd.nist.gov/vuln-metrics/cvss

The CVE contains vulnerabilities and their descriptions. It provides defenders with timely information about the weaknesses of the technologies they use. The CVSS allows the severity classification of identified vulnerabilities.

Unfortunately, these and similar databases are not only used by defenders, but also by cybercriminals, who either purchase software and scripts from others or develop them themselves to exploit newly identified vulnerabilities. Criminals

Table 2.2: Adversary types as defined in NISTIR 7628

Adversary	Description (as defined in NIST IR 7628)
Nation states	State-run, well organized and financed. Use foreign service agents to gather classified or critical information from countries viewed as hostile or as having an economic, military or a political advantage.
Hackers	A group of individuals (e.g., hackers, phreakers, crackers, trashers, and pirates) who attack networks and systems seeking to exploit the vulnerabilities in operating systems or other flaws.
Terrorists or cyberterrorists	Individuals or groups operating domestically or internationally who represent various terrorist or extremist groups that use violence or the threat of violence to incite fear with the intention of coercing or intimidating governments or societies into succumbing to their demands.
Organized crime	Coordinated criminal activities including cyber extortion, IP theft and other. Organized and well-financed criminal organization.
Other criminal elements	Other facets of the criminal community, which is normally not well organized or financed. Normally consists of a few individuals, or of one individual acting alone.
Industrial competitors	Foreign and domestic corporations operating in a competitive market and often engaged in the illegal gathering of information from competitors or foreign governments in the form of corporate espionage.
Disgruntled employees	Angry, dissatisfied individuals with the potential to inflict harm on critical infrastructures. This can represent an insider threat depending on the current state of the individual's employment and access to the systems.
Careless or poorly trained employees*	Those users who, either through lack of training, lack of concern, or lack of attentiveness pose a threat to a CI. This is another example of an insider threat or adversary.

Table 2.3: The five levels of likelihood

Level	Description
Very low	No known previous similar attacks. Cyberattackers are not motivated. No known vulnerabilities.
Low	Vulnerabilities exist that are difficult to attack. There were no such attacks within the sector/industry. Cybercriminals" motivation is low.
Possible	Known vulnerabilities exist in the system. Cybercriminals motivation is moderate. There were similar attacks in other sectors.
Probable	There are known vulnerabilities, but attacks are difficult to automate. Cybercriminals are motivated. There were similar attacks in the sector/industry in the past.
Inevitable	There are known vulnerabilities without effective countermeasures and tools to effectively exploit them. Cybercriminals are highly motivated.

also trade vulnerability descriptions on online marketplaces. Zero day vulnerabilities are usually the most valuable, as they are not yet known to the producer of the affected product, not listed in the CVE database and there is no patch or other tested workaround yet.

2.4.4 Likelihood

We estimate the likelihood on a five-point scale based on the motivation of threat sources, known previous attacks within the sector/industry, and known vulnerabilities of the system under investigation. It describes how likely a successful attack is, not how often different cyber attackers will unsuccessfully probe the system with such an aim (e.g. several times a day, once a year). With this in mind, the probability levels are described in Table 2.3.

It is important to add here that in the field of highly critical sectors, it must be assumed that there are always highly capable and motivated adversaries with the appropriate knowledge to launch a successful attack, i.e. those sectors must assume that a cyberattack is inevitable (its probability is 100%) (Bochman and Freeman, 2021).

Table 2.4: The five levels of impact

Level	Description
Negligible	Negligible material damage. No reputational damage. Can be dealt with in-house. No penalties. No environmental impact. No equipment damage. No human injury.
Low	Minor financial damage. Easily manageable reputational damage.
Medium	Uncomfortably high financial damage and/or penalties. Expensive external experts must be hired. Minimal environmental impact.
Major	Significant financial and reputational damage. Manageable penalties. Limited environmental impact. No human injury. Minor equipment damage.
Catastrophic	Human injury. Significant environmental impact. Major equipment damage. Financial damages or penalties that cannot be covered from own resources, leading to bankruptcy. Total loss of reputation.

2.4.5 Impact

The impact of unwanted events is also sector-dependent.

The impact of a theoretical attack on a water management system can be personal data breach, material damage, damaged equipment in the physical space, and in the worst case, the illness or even death of users consuming water. In addition to these impacts, system operators in regulated sectors must pay penalties if they fail to provide sufficient quality of service (e.g., unplanned electricity outage longer than 3 minutes), experts must be employed to prevent damage, there can be environmental impacts (e.g. sewage spills), and a successful attack can also have a negative impact on the reputation of the organization. Given these, Table 2.4 provides a guide to estimating the impact on a five-level scale.

2.4.6 Risk and risk matrix

As we showed, both likelihood and impact are usually assessed on qualitative levels consisting of three to (usually) five levels. The five levels of likelihood and impact can be combined into a so-called risk matrix as shown in figure 2.6.

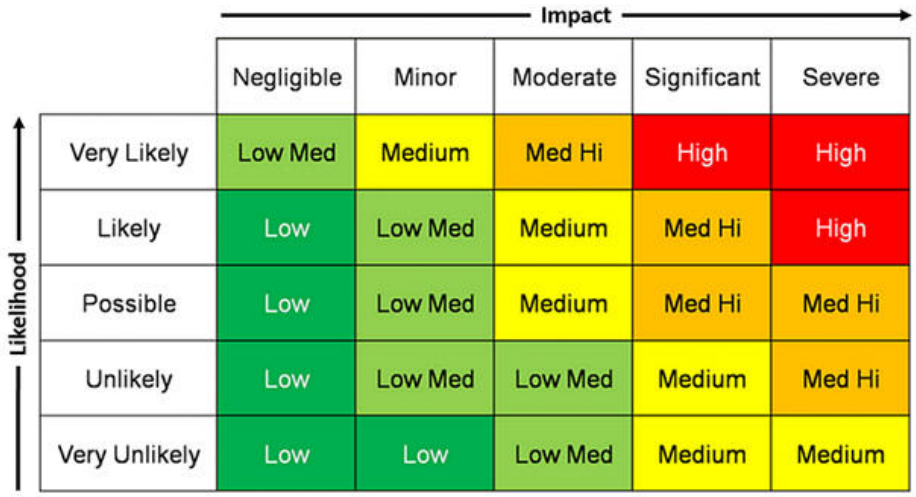


Figure 2.6: Risk matrix (impact and likelihood)

The risk of identified adverse events can be determined using the risk matrix. The matrix is sector- and within them organization-dependent, i.e. it must be determined and updated for each organization during the risk management process based on the latest sectoral and general cybersecurity trends and known attacks. In large organizations with different business lines supported by separate information systems it might be a good idea to maintain different risk matrices for each sybsystem.

After describing each identified potential incident (e.g., cyber attack) and estimating its probability and impact, the appropriate risk level can be read from the organizational (or subsystem) risk matrix. Knowledge of the risks allows for the targeted development and deployment of security controls (i.e., countermeasures) by selecting and tackling the highest risks first (located in the red and orange part of the risk matrix).

Bibliography

Bochman, A. A. and Freeman, S. (2021). *Countering Cyber Sabotage: Introducing Consequence-Driven, Cyber-Informed Engineering (CCE).* CRC Press; 1st Edition.

Bodungen, C., Singer, B., Shbeeb, A., Wilhoit, K., and Hilt, S. (2016). *Hacking*

Exposed – Industrial Control Systems: ICS and SCADA Security Secrets & Solutions. McGraw-Hill Education.

European Parliament and European Commission (2016). General Data Protection Regulation.

European Parliament and European Commission (2022). NIS 2 Directive.

Fennelly, L. (2013). *Effective physical security*. Butterworth-Heinemann.

Tiszolczi, B. (2019). Fizikai biztonsági kontrollok tervezésének és alkalmazásának gyakorlata az ISO/IEC 27001 szabvány elvárásainak tˊukrében. *Magyar Rendészet*, 2019/2—3:233—249.

United States of America (USA) (1974). Privacy Act.

Uptime Institute (2025). Tier Classification System.

Wikipedia (2013). Metcalf sniper attack.

Wikipedia (2020). 2020 United States federal government data breach (Solarwinds).

Chapter 3

Input data manipulation

The purpose of input data manipulation (OWASP ML01:2023) varies and depends primarily on the task performed by the attacked model and the information system that includes it. Researchers primarily focus on attacks against classification tasks, and within that, evasive attacks against image classification solutions.

3.1 Evading spam detection

The first such attacks against solutions known in the field of cybersecurity that also use machine learning were carried out against tools developed for filtering spam Dalvi et al. (2004). Cybercriminals who make a living from spam have used various techniques to bypass filters, e.g. intentional typos, the use of similar characters, inserting text as images, or inserting large amounts of text that is invisible to the user reading the email but disrupts the classification model. The solutions of large IT service providers have developed a lot in this area in the previous period and spam that gets through the filters is orders of magnitude less of a problem than in the early 2000s.

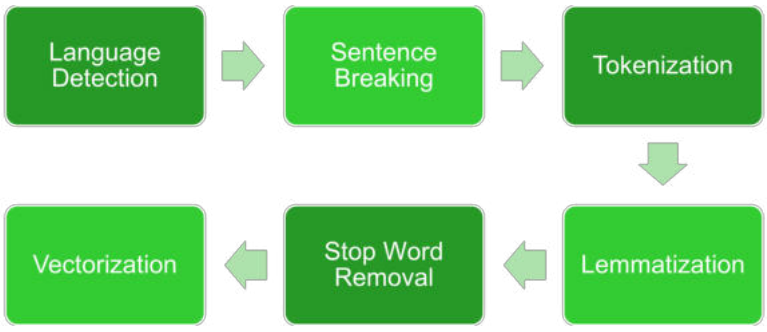


Figure 3.1: Text mining phases

Spam filters are binary text classification systems with two possible decisions: spam or ham (harmless content). Modern text mining solutions usually operate in the phases shown in Figure 3.1. The adversaries which intend to mislead spam filters can and do attempt to target one or more of these phases. Proper language detection can be avoided by mixing text in multiple languages. Sentence breaking algorithms can be misled by inserting unnecessary punctuation marks, sometimes even in the middle of words (e.g., f!lter instead of filter). The misuse of punctuation marks and spaces can similarly mislead tokenization algorithms as well. If any of the first three text mining phases results in errors or partial solutions, then subsequent phases will not execute properly, as it might not be possible to properly reduce words to their base or dictionary form in lemmatization or remove stop words. The resulting vector encodings of the original text might have entirely different meanings and might lead to classification errors. Google developed the RETVec multi-language anti-spam tool[1] which solves most of the here-listed challenges and is able to properly vectorizes text with all the above tricks deployed. It supports over 100 languages and is resilient to homoglyphs (similar characters), intentional typos, deletion or insertion of characters and LEET speak (e.g., T3nn1sP!4y3r).

The criminals pushing spam campaigns usually observe the AI-based spam filter as a black box and attempt to mislead the classification engines without exact knowledge of the underlying AI models, training data, training algorithms and/or hyperparameters.

3.2 Evading image classification

Since the mid-2010s, researchers and the public interested in AI security have been increasingly interested in the misdirection of various image recognition solutions. Among these, we highlight the misdirection of smart cars recognizing traffic signs by labeling the signs in physical space, and the addition of noise invisible to the human eye to images containing various objects and animals, which causes otherwise excellent image recognition solutions to make completely wrong decisions. Attacks can be roughly grouped according to whether (1) the object of classification is modified in physical or electronic space, and (2) the modification is visible or invisible to the human eye (visible label vs invisible noise). In the field of image modification, 1-pixel attacks are interesting, which modify only one pixel in the image and thereby succeed in misleading the AI model.

3.2.1 White-box attacks

In white-box attacks the attackers possess extensive knowledge about the target AI-enabled system. They either hacked into the system hosting the AI model or they obtained insider access. Alternatively, the model and/or training data are open-source and thereby accessible to the attacker.

[1] https://github.com/google-research/retvec

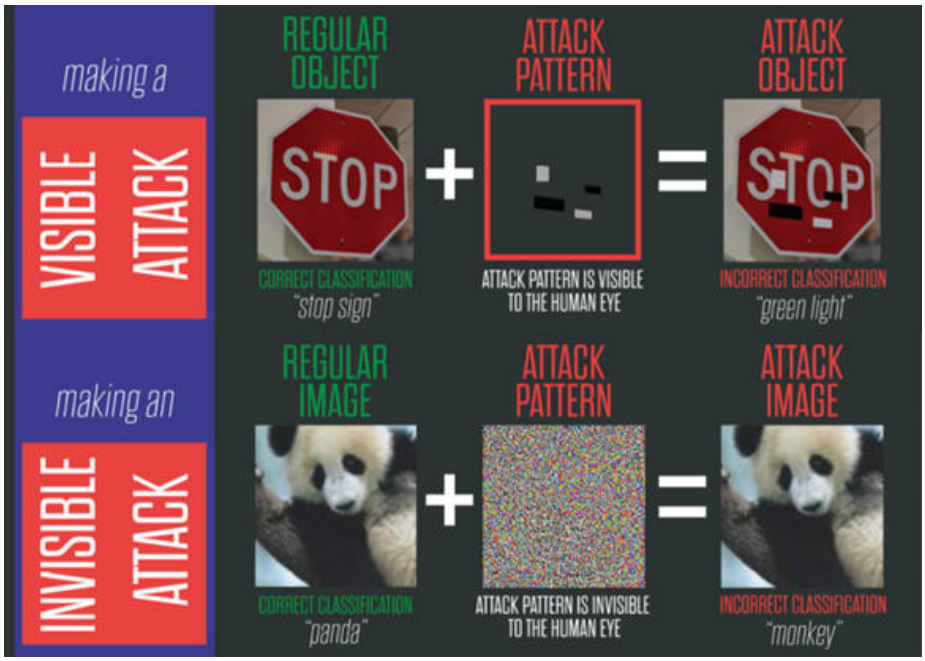


Figure 3.2: Visible vs invisible evasion attack Kotyan (2023).

These information allow the adversary to craft effective input perturbations which cause the model to make incorrect decisions (e.g., misclassify an image with high confidence), even if the modifications are imperceptible to human observers or automated input manipulation detection solutions alike. Attackers calculate the gradient of the model's loss function with respect to the input image, which allows them to determine the direction in which to alter input data (e.g., pixels in images) to maximize the classification error for a given input (Goodfellow et al., 2014). The direct feedback mechanism measured via the gradient allows for the generation of minimal perturbations which efficiently mislead the AI model.

We hereby provide a non-exhaustive list of promising white-box attack methods against image classification systems:

- The single-step **Fast Gradient Sign Method (FGSM)** was introduced by Goodfellow et al. (2014). It computes the gradient of the loss with respect to the input image and then adds a small perturbation to each pixel based on the sign of that gradient, moving the input in the direction that maximally increases the loss. It is computationally efficient but often generates perceptible perturbations.

- The **Basic Iterative Method (BIM) / Iterative FGSM (I-FGSM)** is an extension of FGSM. It applies the FGSM step iteratively with a

smaller perturbation at each step, projecting the perturbed image back into an ϵ-ball (a defined boundary for the perturbation magnitude) after each iteration (Kurakin et al., 2018). This iterative process typically yields stronger adversarial examples which are often less perceptible to the human eye. Projected Gradient Descent (PGD) performs the same steps but initializes to a random point in the ball of interest (Madry et al., 2017).

- The **Carlini & Wagner (C&W)** attack is known for generating highly effective and minimal adversarial perturbations. Unlike gradient-sign-based methods, it uses optimization to find the smallest possible perturbation that leads to misclassification, often resulting in imperceptible changes (Carlini and Wagner, 2017).

3.2.2 Black-box attacks

Black-box attacks are similar to real-world scenarios in which the attackers only interact with the model via its API or observed outputs (Goodfellow et al. (2014)). They do not have access to the model architecture and parameters, neither the training data. To make matters worse (for the attacker), the interface via which they communicate with the target AI model usually is rate-limited, which means that the number of queries submitted in a unit of time is limited. The fundamental challenge in black-box attacks is the absence of gradient information, which is central to white-box methods discussed above.

Even in such constrained environments, attackers usually possess limited knowledge about the model architecture, training algorithm and hyperparameters, based on which they might be able to identify optimal adversarial data by generating and testing a small number of samples.

Black-box attackers' approaches fall into the following main categories: transferability and query-based attacks. Figure 3.3 illustrates a recent, non-exhaustive and somewhat similar taxonomy of black-box adversarial attacks.

Attack transferability

The goal of a transfer attack is to generate and test malicious input data on a (substitute or shadow) model developed or acquired by the attacker and then transfer the tested adversarial data and use it against a target model. It may seem a bit surprising that such attacks can work at all, especially in a black-box attack in which the attacker has no access to either the training data or the target AI model itself, but various researchers have demonstrated the feasibility of these attacks on different models (Demontis et al. (2019); Yin et al. (2023)).

The transfer attack can be explained by the similarities of the training sets and the general weaknesses of AI models. If two AI models with similar architectures are trained using similar training sets, they are expected to learn similar decision planes and make incorrect decisions along them on similar adversarial patterns. A general weakness is when the model identifies and learns features or relationships (correlations) between them as important during training that

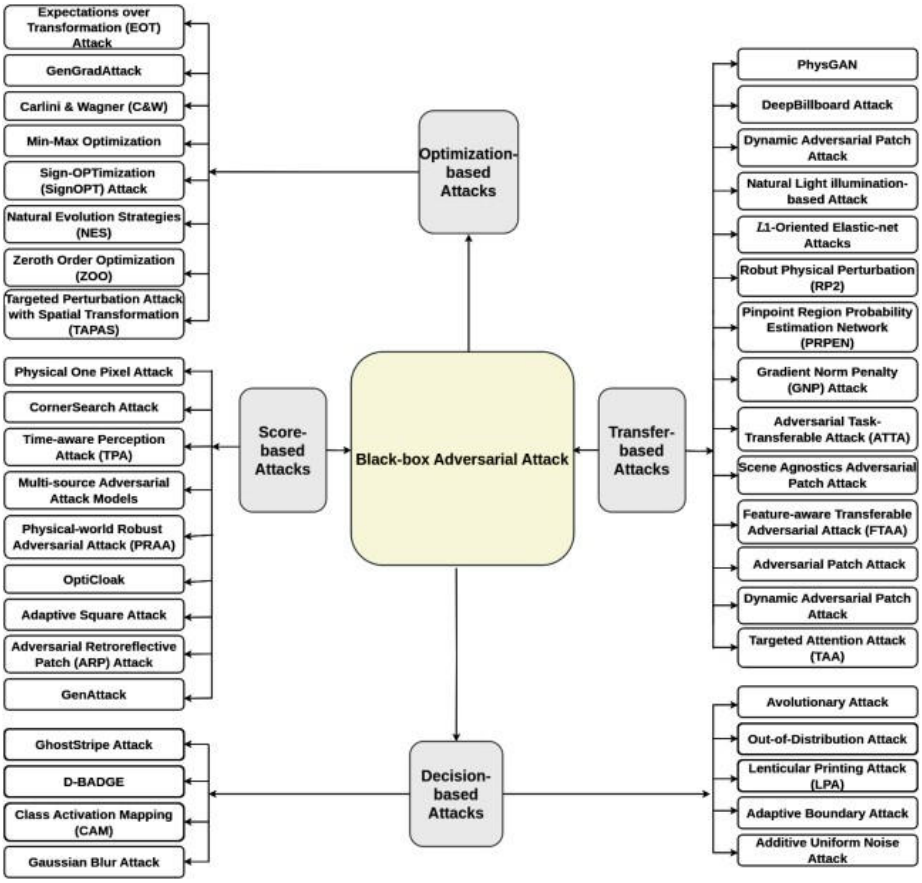


Figure 3.3: Taxonomy of black-box adversarial attack against image classification models Badjie et al. (2024).

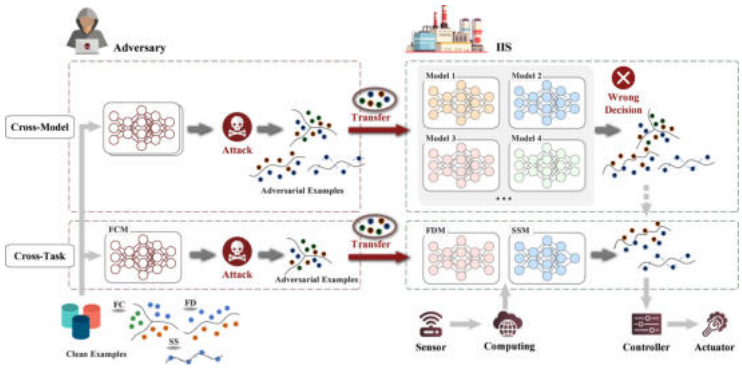


Figure 3.4: Transferability of manipulated input data Yin et al. (2023).

are otherwise not significant in the given domain. Sophisticated attackers can base their input data manipulation strategies on their knowledge about such suboptimal learned patterns.

Query-based Attacks

Query-based attacks iteratively perturb inputs and query the target AI model for its decisions (e.g., predictions). By observing changes in confidence scores or predicted labels, the attacker might infer information about the model's decision boundaries. These attacks are usually computationally expensive due to the large number of queries required, and as such they tend to trigger rate limiting-based detection mechanisms.

The general steps performed by the query-based attacker are usually the following:

- (Optional, if highly skilled attacker) Design attack.
- Generate limited set of input data samples which maximize knowledge gain about the model once submitted to the model and paired with the resulting outputs.
- Submit adversarial samples to the target AI model .
- Collect and analyze outputs. Observe confidence scores and other model performance indicators to further optimize adversarial sample generation.
- Generate adversarial samples which will be utilized to reach the attackers' objectives on the target model.

In optimization-based query attacks the attacker's objective is to identify the smallest possible perturbation which could lead to incorrect AI model decisions. The attacker iteratively modifies the input in order to maximize the model's

standard loss or reduce its confidence in the actual class while ensuring that the alteration remains undetectable.

Decision boundary attacks (DBA) exploit machine learning models by finding tiny changes (perturbations) to an input that push it across the model's decision boundary, causing misclassification. Such attacks rely only on the known model output (label) and can be deployed without gradient access (Jin et al., 2022).

Query-based attacks focusing on gradient estimation approximate the gradient of the loss function with respect to the input, based on the model's output scores[2]. The steepest ascent of the loss function can be approximated by querying the model with nearby points i.e., inputs corresponding to nearby points in the input space(Li et al., 2021). The assessed points can be selected by exhaustively checking each neighboring point along each dimension of the input space, or by a guided random search. Gradient estimation approaches often require fewer queries to identify an initial adversarial example compared to decision boundary approaches.

Bibliography

Badjie, B., Cecílio, J., and Casimiro, A. (2024). Adversarial attacks and countermeasures on image classification-based deep learning models in autonomous driving systems: A systematic review. *ACM Comput. Surv.*, 57(1).

Carlini, N. and Wagner, D. (2017). Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57. Ieee.

Dalvi, N., Domingos, P., Mausam, Sanghai, S., and Verma, D. (2004). Adversarial classification. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 99–108. Association for Computing Machinery (ACM).

Demontis, A., Melis, M., Pintor, M., Jagielski, M., Biggio, B., Oprea, A., Nita-Rotaru, C., and Roli, F. (2019). Why do adversarial attacks transfer? explaining transferability of evasion and poisoning attacks. In *28th USENIX security symposium (USENIX security 19)*, pages 321–338.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014). Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Jin, H., Chen, J., Zheng, H., Wang, Z., Xiao, J., Yu, S., and Ming, Z. (2022). Roby: Evaluating the adversarial robustness of a deep model by its decision boundaries. *Information Sciences*, 587:97–122.

[2]https://apxml.com/courses/adversarial-machine-learning/chapter-2-advanced-evasion-attacks/score-based-attack-techniques

Kotyan, S. (2023). A reading survey on adversarial machine learning: Adversarial attacks and their under-standing. *arXiv preprint arXiv:2308.03363*.

Kurakin, A., Goodfellow, I. J., and Bengio, S. (2018). Adversarial examples in the physical world. In *Artificial intelligence safety and security*, pages 99–112. Chapman and Hall/CRC.

Li, H., Li, L., Xu, X., Zhang, X., Yang, S., and Li, B. (2021). Nonlinear projection based gradient estimation for query efficient blackbox attacks. In *International conference on artificial intelligence and statistics*, pages 3142–3150. PMLR.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. (2017). Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

Yin, Z., Zhuo, Y., and Ge, Z. (2023). Transfer adversarial attacks across industrial intelligent systems. *Reliability Engineering & System Safety*, 237:109299.

Chapter 4

Confidentiality attacks

The attacker's goal in confidentiality attacks is to steal intellectual property. In applied AI systems that is most frequently sensitive training data, or key artifacts utilized or produced in the training phase, namely serialized AI models, training algorithms, or specific hyper parameter values.

We first focus on model inversion and then discuss other confidentiality attacks.

4.1 Model inversion

AI model inversion attacks are a significant privacy threat in machine learning, aiming to reconstruct sensitive attributes or even entire training data samples by exploiting access to a trained model (Fredrikson et al., 2015). Unlike input manipulation attacks which seek to alter decisions made by the AI model, inversion attacks aim to reverse-engineer parts of the training dataset (OWASP ML03:2023). This becomes a major concern when AI models are trained on highly sensitive personal information (facial images, medical records, genomic data) or sensitive corporate data (financial data, annual reports, production recipes).

The core principle behind model inversion is that ML models, particularly highly accurate models, inherently memorize certain aspects of their training data to generalize effectively (Shokri et al., 2017). An attacker, by observing the model's outputs (e.g., confidence scores, feature embeddings, or even just class predictions), attempts to deduce these underlying training characteristics.

Fredrikson et al. (2015) demonstrated the feasibility of model inversion against logistic regression models trained on sensitive data. They showed that an attacker with knowledge of non-sensitive attributes of a target individual, could infer their sensitive attributes (e.g., medical conditions) by querying a personalized medicine model.

The evolution of generative models, particularly Generative Adversarial Networks (GANs), significantly enhanced the power of model inversion attacks. Re-

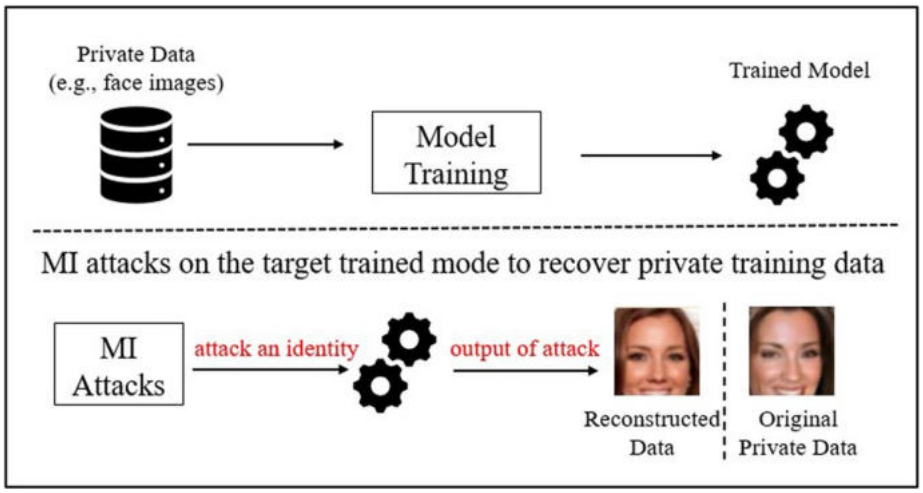


Figure 4.1: A visualization example of the model inversion (MI) attack Yang et al. (2025).

searchers leveraged GANs to learn a prior distribution of real-world data from publicly available datasets. This prior then guided the inversion process, allowing attackers to generate more realistic and higher-fidelity reconstructions of private training data by optimizing an input to the generative model that produces outputs matching the target model's behavior for a specific class (Zhang et al., 2020).

In general, model inversion attacks pose substantial privacy risks, including:

1. **Personal data exposure.** Direct reconstruction of sensitive personal identifiable information (PII) like faces or medical conditions. This includes non-compliance with data protection laws like GDPR, which emphasize data minimization and privacy by design.

2. **Trade secret exposure.** Inferring proprietary data or processes that might have been part of a model's training set. For example, a model trained on source code examples might leak a proprietary algorithm developed by a company whose source code was included in the training set.

Attackers face a significantly more difficult task if they only have black-box access to the AI system i.e., do not have direct access to the model and cannot analyze its responses in a laboratory environment. If in addition there is rate-limiting on the API or web interface used to query the model, then the attackers must be very clever and ask the AI system optimally chosen questions, observe the answers received and based on them infer the elements of the training set.

4.2 Membership inference

In membership inference attacks an adversary aims to determine whether a specific data point was part of the model's training dataset (Shokri et al., 2017a). The attacker possesses at least partial knowledge of the data point (e.g., one cancer patient's basic information like name, surname, year of birth) and their intention is to check whether the known record was included in the training set? This type of attack exploits the phenomenon of memorization, where models, especially over-parameterized deep neural networks, implicitly "remember" certain characteristics of their training data to generalize effectively.

The presence of a certain, partially known record in a training set is not necessarily an issue, unless the attacker can also infer the values of sensitive attribute (e.g., has cancer, credit rating or similar). Attackers are usually motivated to build attack/classifier to check if certain private attribute has a certain value(s) for the target record. If successful, such an attack can compromise the privacy of individuals whose data (e.g., medical records, financial transactions) or other sensitive data (e.g, corporate data) was used in training, even if the data itself was not explicitly leaked.

The methodology presented by Shokri et al. (2017b) involved training surrogate (or shadow) models which mimic the behavior of the target model. Each shadow model is trained on a subset of data that is a "member" of its training set and a subset that is "not a member." By observing the target model's behavior (e.g., prediction probabilities or confidence scores) on a given data record, an attacker can then train an attack model to distinguish between members and non-members, based on the patterns observed from the shadow models. For instance, models tend to output higher confidence scores for data points included in their training sets compared to unseen data.

The effectiveness of membership inference attacks is often linked to the model's overfitting (Yeom et al., 2018). Models that overfit essentially better memorize their training data and therefore tend to exhibit a more distinct behavior when an element of their training dataset is submitted to their inputs in the inference/operations stage (as compared to non-members). This characteristic makes them more vulnerable to membership inference attacks. Yeom et al. (2018) showed that a model's susceptibility to membership inference can be quantified by analyzing its loss values on training versus testing data. More specifically, lower loss on a data point typically indicates it was a member of the training set.

Nasr et al. (2018) conducted a more comprehensive analysis of privacy vulnerabilities, highlighting how models leak information about their training data distribution beyond mere membership. This work further solidified the understanding of how model outputs (such as activation patterns or layer outputs) can implicitly reveal membership. Further investigations have shown even raw gradients can leak membership, as demonstrated by Zhu et al. (2019), revealing that sensitive training data can be reconstructed directly from shared gradients in distributed learning settings.

In summary, we showed that by examining the model available to the at-

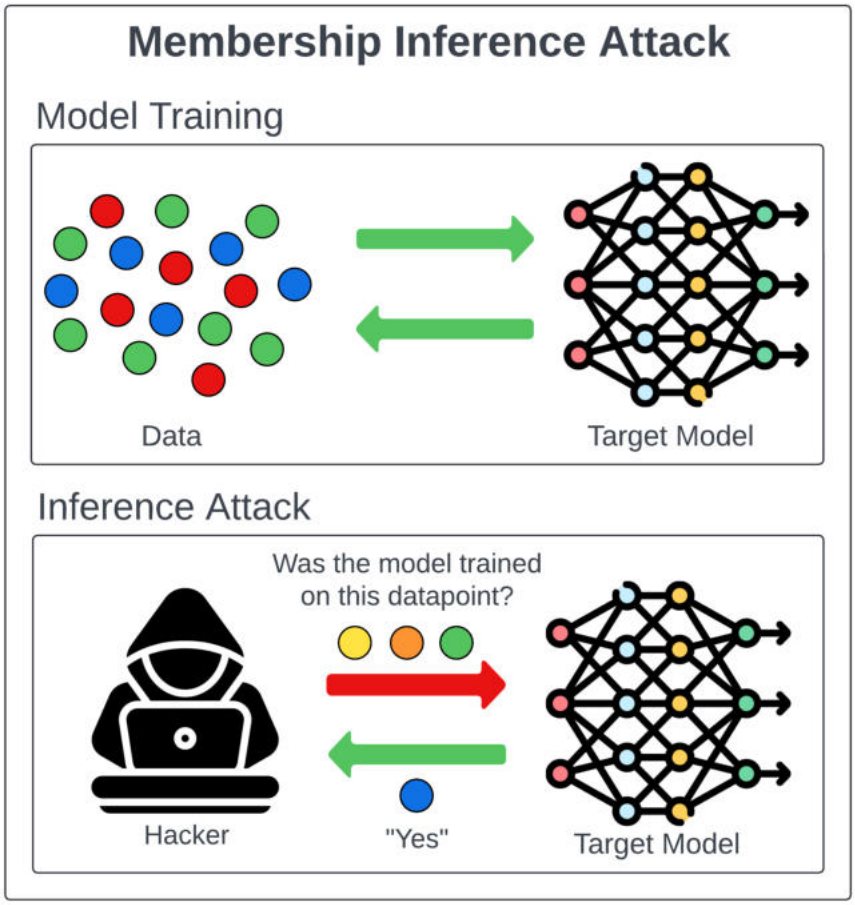


Figure 4.2: Membership inference attack overview Mindgard (2025).

tacker, or in the case of a black box attack, based on the answers to a larger number of questions, the confidentiality of the training data may be compromised if the attacker can successfully prove, based on his investigations, whether a given data item was part of the training set or not e.g., if the training data included a data row describing an illness associated with a human. This attack is identified by the OWASP Top 10 list as ML04:2023.

4.3 Model theft

In model theft attacks (OWASP ML05:2023), also known as model extraction or model stealing, an adversary illicitly obtains a functional copy or an approx-

imation of a proprietary machine learning (ML) model (Tramèr et al., 2016). Unlike model inversion (which targets the training data), these attacks target the intellectual property embedded within the trained model itself, potentially compromising significant investments in research, development, and data collection. The primary motivation for such attacks often includes gaining a competitive advantage, bypassing licensing fees, or facilitating subsequent adversarial attacks.

Trivial model theft scenarios involve obtaining unauthorized access to information systems hosting copies of serialized AI models and making unauthorized copies of them. Such theft is possible if the adversary manages to hack into the computer networks used for storing serialized models, or if the adversary manages to convince malicious insiders to leak models.

More advanced adversaries might take an approach similar to query-based black-box attacks. Tramèr et al. (2016) demonstrated that an attacker, with only query access to a black-box model (i.e., observing inputs and outputs without knowing its internal architecture or parameters), could construct a substitute model that closely mimics the target model's functionality. This is achieved by generating a synthetic dataset of queries and recording the target model's responses. The attacker then trains their own model (the "substitute" or "stolen" model) on this synthetic dataset. If the target model behaves deterministically and consistently, even a relatively small number of queries can enable the creation of a highly accurate replica.

In active learning attackers select "highly-informative" queries, by creating input samples close to the decision boundary, which are "hard-to-classify" and result in model decisions with low confidence. The main idea is that these samples help the most with localizing the decision boundary and thereby extracting the model itself.

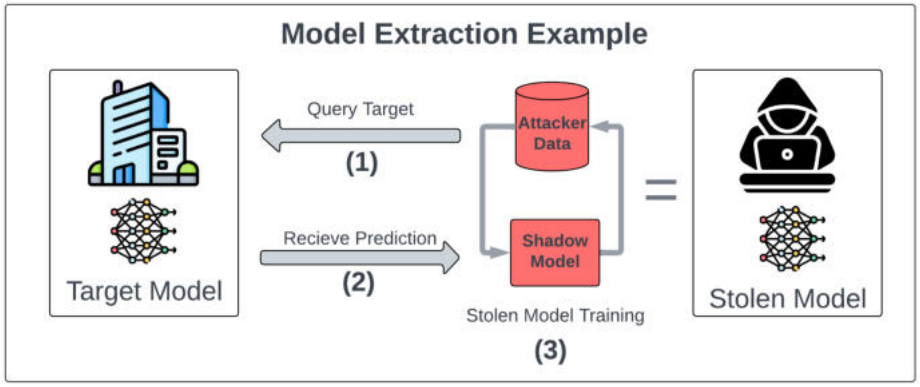


Figure 4.3: Model extraction overview Mindgard (2025).

We must also add that some AI models can be of significant financial value due to the sensitive data they contain and/or the significant resources spent on training them. The high value increases the motivation of cybercriminals to steal

such models and sell them on the black market or otherwise obtain unsolicited revenue. Model inversion/extraction can be a stepping stone for additional, typically black-box attacks which need surrogate models, e.g., model inversion, membership inference, evasion attacks.

Bibliography

Fredrikson, M., Jha, S., and Ristenpart, T. (2015). Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1322–1333.

Mindgard (2025). Ai under attack: Six key adversarial attacks and their consequences.

Nasr, M., Shokri, R., and Houmansadr, A. (2018). Comprehensive privacy analysis of deep learning. In *Proceedings of the 2019 IEEE Symposium on Security and Privacy (SP)*, volume 2018, pages 1–15.

Shokri, R., Stronati, M., Song, C., and Shmatikov, V. (2017a). Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18. IEEE.

Shokri, R., Stronati, M., Song, C., and Shmatikov, V. (2017b). Membership inference attacks against machine learning models.

Tramèr, F., Zhang, F., Juels, A., Reiter, M. K., and Ristenpart, T. (2016). Stealing machine learning models via prediction apis. In *25th {USENIX} Security Symposium ({USENIX} Security 16)*, pages 601–618.

Yang, W., Wang, S., Wu, D., Cai, T., Zhu, Y., Wei, S., Zhang, Y., Yang, X., Tang, Z., and Li, Y. (2025). Deep learning model inversion attacks and defenses: a comprehensive survey. *Artificial Intelligence Review*, 58(8):242.

Yeom, S., Giacomelli, I., Fredrikson, M., and Jha, S. (2018). Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, pages 268–282. IEEE.

Zhang, Y., Jia, R., Pei, H., Wang, W., Li, B., and Song, D. (2020). The secret revealer: Generative model-inversion attacks against deep neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 253–261.

Zhu, L., Liu, Z., and Han, S. (2019). Deep leakage from gradients. *Advances in neural information processing systems*, 32.

Chapter 5

Integrity attacks

Artificial Intelligence (AI) system integrity attacks aim to alter one or more elements of said system, usually the training (or validation or tuning) dataset, the algorithm, the hyperparameters used, the model architecture or parameters or its outputs. Upon sucessful integrity violations the AI system can no longer be relied upon to perform its intended function correctly or consistently, potentially leading to errors, biased decisions, malicious or other unintended behavior.

Of the above listed potential targets of integrity attacks, we focus our attention on data and model poisoning, the poisoning of data and AI models, as well as supply chain attacks against AI systems.

5.1 Data poisoning

Data poisoning (OWASP ML02:2023) attacks occur during the training phase of an AI model and they involve either targeted or general unauthorized modifications of the training data. The adversaries inject corrupted or carefully crafted data into the training dataset to compromise the model's integrity and performance. The attacker can modify a subset of the data or the entire data set if the goal is to deliberately skew the statistical distribution of one or more features.

Biggio et al. (2012) formally introduced poisoning attacks against support vector machines (SVM), demonstrating that an attacker could strategically add a small number of adversarial training samples to cause misclassification errors at test time. This work highlighted the vulnerability of learning algorithms to manipulated training data. Subsequent research extended these principles to deep neural networks, where the scale and complexity of models present new challenges and opportunities for attackers (Guo and Liu, 2020).

Data poisoning attacks can be broadly categorized by their objective. Targeted attacks aim to cause misclassification of specific target inputs while leaving the model's performance unaffected for other inputs. An example might involve injecting data to ensure a particular face is always misidentified. Indiscriminate

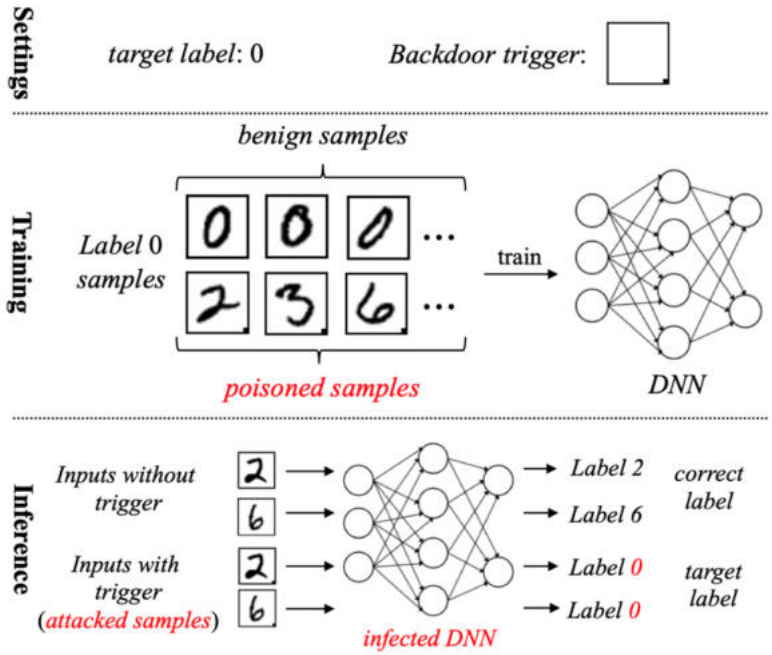


Figure 5.1: Backdoor insertion with data poisoning - the dot in the lower right corner is the trigger (https://velog.io/@tjdcjffff/Backdoor-Attack)

(aka availability) attacks are different, as they focus on degrading the overall accuracy or availability of the model, rendering it less useful or unusable. This could involve introducing noisy or conflicting labels across a large portion of the training data.

The attacker's goal can also be to insert backdoors i.e., modify the training data in such a way that the model trained model always returns a given answer or makes a pre-defined decision when it receives input data containing a specific pattern (Li et al., 2023). A model trained on poisoned data might correctly identify all objects except when a specific, imperceptible pattern is present and causes a malicious misclassification. An example of this can be seen in Figure 5.1, in which the attacker modifies the training set by inserting a dot in the lower right corner. Later, when the attacker sends an input marked with a dot to the running AI system after deployment, the model is very likely to make the decision expected by the attacker.

Since the attacker must have write access to the training data set, this type of attack is usually white or at least gray box.

Apart from direct data poisoning, the attacker might also choose more covert means of poisoning other artifacts of the model training phase. They might

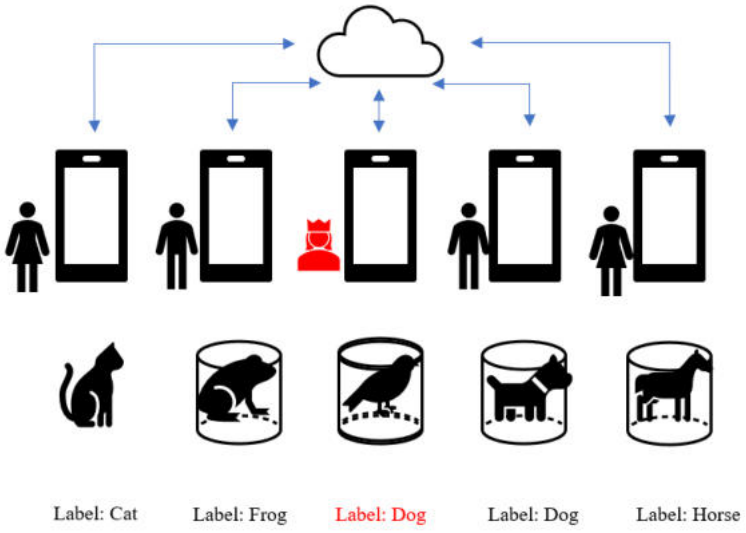


Figure 5.2: Label flipping in data poisoning

attempt to alter the training algorithm or hyperparameters used during the training phase, as well as the parameters of a trained model itself.

5.1.1 Data poisoning classification systems

Label flipping and clean-label data poisoning represent two distinct strategies for achieving an adversary's malicious objectives.

Label Flipping Attacks

In label flipping attacks an adversary directly alters the class labels of a subset of training data points and thereby steers the model's (trained) decision boundary in a desired direction, often to cause misclassification of specific target samples at inference time, or to degrade the model's overall performance (Biggio et al., 2012). The manipulated data points retain their original feature values, but their associated labels are flipped to an incorrect class.

For example, in a binary classification task of "cat" vs. "dog," an attacker might change the label of some "cat" images to "dog" and vice-versa (see Figure 5.2). The attacker's objective might be targeted in which only certain, high-value inputs are misclassified, or they might attempt to generally confuse the model, reducing its overall accuracy across multiple classes.

The effectiveness of label flipping often depends on the adversary's knowledge of the target model's architecture or the training process. Techniques often involve identifying "vulnerable" training samples near the decision boundary

first, which if mislabeled, exert maximum influence on the model's learning for untargeted attack. For targeted attacks, the attacker might surround the target record with mislabeled instances that are similar (in the feature space) to the target instance.

Executing a label flipping attack is challenging as it is relatively easy to detect either with anomaly detection (e.g., trained model makes incorrect prediction on mislabeled samples, which tend to have large loss) or with a human annotator.

Clean-Label Attacks

In clean-label data poisoning attacks the adversary injects poisoned data points into the training set that retain their original, correct labels, but their features are subtly perturbed (Shafahi et al., 2018). The malicious intent is embedded entirely within the feature space of the poisoned samples, making them appear "clean" and legitimate to human inspection or automated label-checking mechanisms.

The objective of clean-label attacks can often be to induce a backdoor into the model. This means the model will behave normally for most inputs but will consistently misclassify specific target inputs when they contain a particular, often imperceptible, "trigger" pattern. The poisoned samples are carefully crafted such that their features, when combined with their correct labels, coerce the model to learn an association between the trigger and the desired malicious output (see Figure 5.1).

For example, an attacker might add a small, specific pattern (the trigger) to a subset of "stop sign" images, but these images are still correctly labeled as "stop sign." When the model is deployed, any input image containing this trigger, even if it's a "yield sign," might be classified as a "stop sign." This type of attack is highly realistic because it does not require the attacker to compromise the labeling process, only to inject specially crafted samples. The stealthiness arises from the fact that the labels are correct, making detection difficult.

5.2 Model poisoning

In model poisoning (OWASP ML10:2023), the attacker's goal is to directly or indirectly modify the AI model to achieve their own goals. In a white or gray box attack, the attacker has access to and can modify the architecture and/or parameters of the AI model. While data poisoninig is most relevant in the training phase, model poisoning has high relevance during AI system operations (commonly referred to as the inference phase).

Similarly to data poisoning, model poisoning can also be general or targeted. In general poisoning, the attacker's goal may be the general degradation of model performance. Targeted model poisoning can also insert a backdoor directly into the model by modifying its parameters or architecture in such a way that the

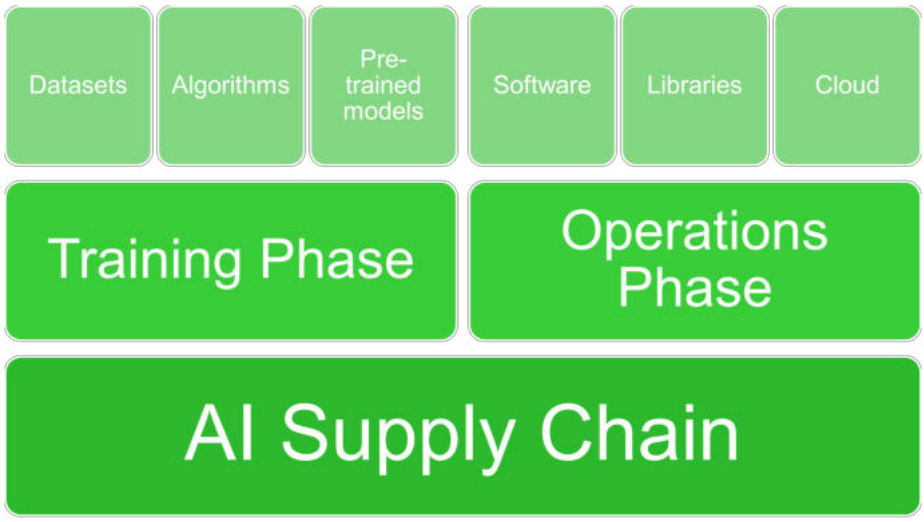


Figure 5.3: Locations of potential vulnerabilities in the AI supply chain

model makes an attacker-designed and desired decision in response to adversarial input data generated by the attacker.

5.3 Supply chain attacks against the AI

Like other information systems, AI systems can be compromised through their supply chains (OWASP ML06:2023), as they can be vulnerable to malicious modifications of the software, libraries, models or data they use (see Figure 5.3 for details). Such vulnerabilities pose a threat both in the training and in the operations phase. This means that the often open-source software used in AI training and operations might be intentionally modified by a highly-skilled adversary. The goal of the attacker is to slightly modify the software or libraries and thereby alter future AI model decision-making processes.

In the training phase of AI models, developers can use publicly available data or models, which can be replaced or poisoned by attackers. Several leading universities and research laboratories maintain their own data repositories (e.g. UCI Machine Learning Repository[1]), and there are also independent data and model sharing systems (e.g. Kaggle[2], Hugging-Face[3]). A malicious attacker might obtain access to such trusted repositories and modify the data/models in order to reach specific future objectives. In a simple example, attackers insert a model backdoor into an open-source model frequently used by others, allowing them to force specific decisions with well-crafted malicious inputs. It

[1] https://archive.ics.uci.edu/
[2] https://www.kaggle.com/
[3] https://huggingface.co/

is important to add, that backdoors/poisons injected in the pre-training phase are preserved even after these pre-trained, foundational models are fine-tuned on local, private data.

Bibliography

Biggio, B., Nelson, B., and Laskov, P. (2012). Poisoning attacks against support vector machines. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ICML'12, page 1467–1474, Madison, WI, USA. Omnipress.

Guo, J. and Liu, C. (2020). Practical poisoning attacks on neural networks. In *European Conference on Computer Vision*, pages 142–158. Springer.

Li, Y., Zhang, S., Wang, W., and Song, H. (2023). Backdoor attacks to deep learning models and countermeasures: A survey. *IEEE Open Journal of the Computer Society*, 4:134–146.

Shafahi, A., Huang, W. R., Najibi, M., Suciu, O., Studer, C., Dumitras, T., and Goldstein, T. (2018). Poison frogs! targeted clean-label poisoning attacks on neural networks. *Advances in neural information processing systems*, 31.

Chapter 6

Availability attacks

Availability attacks against AI systems are those designed to disrupt or prevent the system from performing its intended function, thereby reducing or eliminating its utility. The goal isn't to trick the model into making a specific wrong decision (input manipulation), to extract sensitive information (confidentiality attacks), or to make unauthorized modifications in the AI system (integrity attacks), but rather to make the system inaccessible or so inefficient that it becomes unusable.

6.1 Model-focused Availability Attacks

In model-focused availability attacks the attacker gains access to the wider information system incorporating the AI model and then purposefully modifies the architecture or parameters of the model, making it difficult or impossible for regular users to obtain any or correct results for queries submitted to the AI model. The attacker usually injects "noisy" or contradictory data thereby ensuring that the model fails to converge, or learns a suboptimal ("chaotic") decision boundary that its overall accuracy drops so low it becomes useless in production environments.

6.2 Availability Attacks via Upstream Dependencies

Malicious data, algorithms, models, libraries or software can also be smuggled into the supply chain of an AI system, which can limit is availability once incorporated. For example, the targeted modification of an open-source library used in training can have such consequences if it is altered in such a way that while in use and when triggered via a backdoor or otherwise, it would have significantly degraded performance and thereby make the whole AI system unavailable.

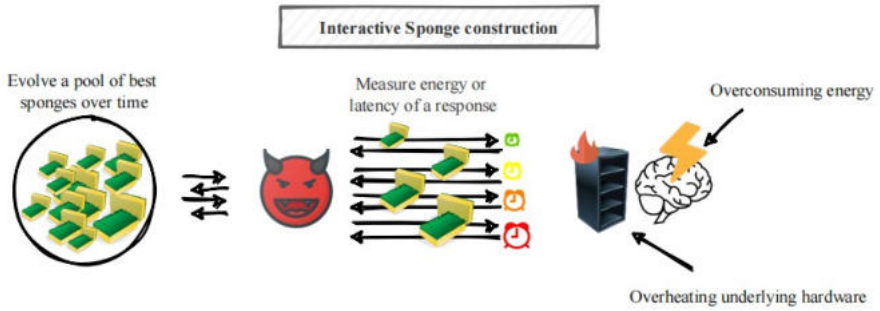


Figure 6.1: Sponge attack overview Shumailov et al. (2021)

6.3 Sponge attacks

Sponge attacks are a type of adversarial attack against AI systems, particularly machine learning models like deep neural networks (DNNs) and Large Language Models (LLMs), that primarily target their availability and efficiency rather than their accuracy. They degrade AI model performance by forcing it to consume disproportionately large amounts of computational resources (e.g., CPU cycles, GPU memory, energy) during inference. This is achieved by crafting specific "sponge examples" – malicious inputs that, while seemingly benign or slightly perturbed, exploit the internal workings of the AI model to trigger computationally expensive operations. If they are executed and not detected on time, they might result in degraded AI system performance and significantly higher utility bills (e.g., electricity or CPU time billed in cloud data centers). In extreme cases, sponge attacks can hog resources and effectively render an AI system unavailable to legitimate users, leading to a denial of service.

Common adversarial tactics include, but are not limited to the following:

1. **Increased Neuron Activation.** Sponge examples can be designed to activate a significantly larger number of neurons in a neural network than typical inputs. More activated neurons mean more calculations, leading to higher energy consumption and latency (Cinà et al., 2025).

2. **Exploiting Algorithmic Complexity.** AI models with adaptive input dimensions or complex internal pipelines can be manipulated into worst-case performance scenarios. For instance, in LLMs, specially crafted text can force the model to process an unusual number of tokens or engage in demanding recursive operations (Gao et al., 2024).

3. **Adversarial Perturbations.** Similarly to traditional adversarial attacks which usually mislead classification systems with slightly modified inputs, sponge attacks can use subtle, noise-like perturbations to "easy" inputs, forcing them to take longer, more resource-intensive paths through the network (Boucher et al., 2022).

4. **Data poisoning.** Sponge attacks can be initiated during the training phase model poisoning. More specifically, malicious data samples can be introduced into the training dataset, which do not affect model performance, but embed vulnerabilities that cause it to become inefficient when encountering specific triggers during operations (Huang et al., 2024).

In the above-listed tactics the attackers exploit the "energy-latency gap" phenomenon, in which AI systems (or any information system) respond to inputs of the same size with vastly different amounts of time and energy consumption. Sponge attacks exploit this by creating inputs that fall into the high-energy and/or high-latency category (Shumailov et al., 2021).

Most, if not all of the above tactics can be executed by highly-skilled adversaries only. Additionally, they can be executed stealthily, leaving minimum traces, while causing significant harm. Sponge examples can be crafted to appear benign or have minimal impact on output accuracy, making them harder to detect through AI system security monitoring which (mainly) focus on correct output Shapira et al. (2023).

A key characteristics of sponge attacks is that they exploit the computational resources of the AI model rather than its logical decision-making process. While less immediate, sustained sponge attacks can contribute to a larger carbon footprint of AI systems due to their excessive energy demands. Additionally, they increase decision-making delay which can be a problem in applications with real-time constraints e.g., in autonomous driving or manufacturing facilities with high levels of automation.

Bibliography

Boucher, N., Shumailov, I., Anderson, R., and Papernot, N. (2022). Bad characters: Imperceptible nlp attacks. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1987–2004.

Cinà, A. E., Demontis, A., Biggio, B., Roli, F., and Pelillo, M. (2025). Energy-latency attacks via sponge poisoning. *Information Sciences*, 702:121905.

Gao, K., Pang, T., Du, C., Yang, Y., Xia, S.-T., and Lin, M. (2024). Denial-of-service poisoning attacks against large language models.

Huang, B., Pang, L., Fu, A., Al-Sarawi, S. F., Abbott, D., and Gao, Y. (2024). Sponge attack against multi-exit networks with data poisoning. *IEEE Access*, 12:33843–33851.

Shapira, A., Zolfi, A., Demetrio, L., Biggio, B., and Shabtai, A. (2023). Phantom Sponges: Exploiting Non-Maximum Suppression to Attack Deep Object Detectors . In *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 4560–4569, Los Alamitos, CA, USA. IEEE Computer Society.

Shumailov, I., Zhao, Y., Bates, D., Papernot, N., Mullins, R., and Anderson, R. (2021). Sponge examples: Energy-latency attacks on neural networks. In *2021 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 212–231.

Chapter 7

Protecting AI systems

Let us assume that the information system encapsulating the AI model implements appropriate security controls e.g. physical and electronic endpoint protection, well-trained workforce, network segmentation, intrusion detection, security monitoring, anti-malware, patching and has a formal backup policy. Even in such systems, the entire life cycle of the AI (sub)system must be protected with additional, AI-specific security controls.

Additionally, one must consider that sophisticated attackers will continuously evolve their attacks (similarly to spammers or malware creators). If we assume that adaptive adversaries possess sufficient knowledge about the state of the art defenses, then we must theorize that they can evade them by generating additionally tailored attacks e.g. further specialized adversarial examples. This turns the challenge into a cat-and-mouse game (or arms race) between attackers and defenders, in which the attackers are usually one step ahead of the defenders.

The goal of this chapter is to present selected techniques which improve the security posture of applied AI systems. We start with data protection, then turn our attention to securing AI models and conclude the chapter AI monitoring.

7.1 Data protection in AI systems

We analyse synthetic data generation (SDG) and differential privacy (DP) as valuable techniques in the toolsets of AI system defenders. SDG is important when foundational or other high-value models are built by third parties which cannot be provided acccess to the otherwise sensitive training data. Differential privacy allows us to alter personally identifiable information (PII) by adding controlled amounts of noise, but maintaining data utility in model training.

7.1.1 Synthetic data generation

One of the major challenges of applied AI systems is the protection of training data. Even if an attacker has access to the attacked AI system as a black box, they can still extract information from the model using model inversion based on the appropriate quantity and quality of queries and the answers received. If the model was trained using highly sensitive data, such as data from the fields of finance or healthcare, it is worth considering replacing the training data entirely with synthetic data.

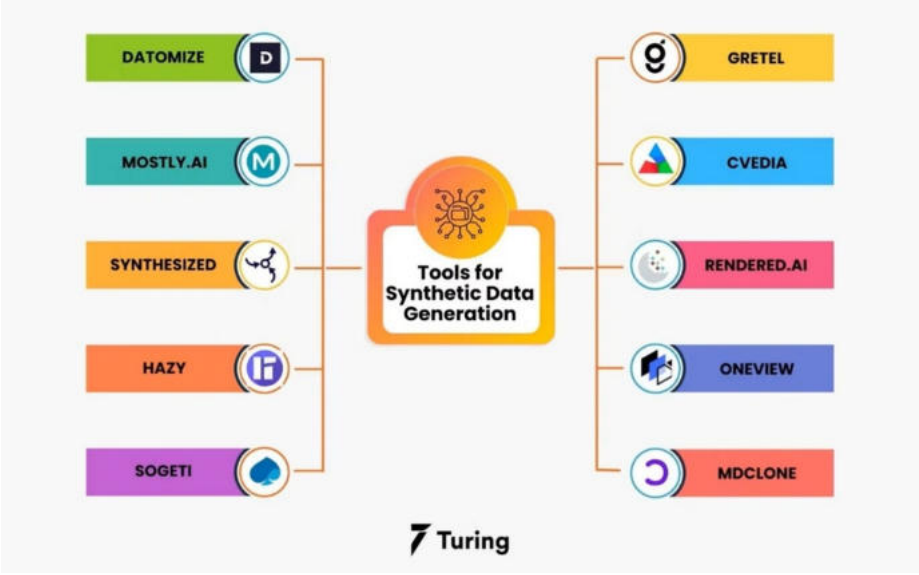


Figure 7.1: Synthetic data generation tools (Turing.com (2022))

Generative AI is capable of generating different types of data (e.g. text, images, videos, time series), but its use is not (always) justified for the formation of synthetic data due to the high (material and energy) costs. Tabular data can be obtained by anonymizing the original data or by applying differential data protection solutions. There are various algorithms that, based on the analysis of the statistical characteristics of the original tabular data (e.g. copula functions), form very similar, completely synthetic data. If we have a sufficient amount of text data, then after vectorizing it, very similar text can be synthesized. Images and videos can also be generated based on the knowledge of their most important features extracted from them. It is important to note that synthetic data also has confidentiality challenges. For example, it is possible that very skilled attackers examine the synthetic data and, based on their examination, conclude which data set the synthetic data was based on, or whether a given data instance was included in the original set. Such data leaks may bear significant similarities to previously reported attacks on AI confidentiality.

7.1.2 Differential privacy

Differential privacy (DP) works by adding a carefully calculated amount of statistical noise (randomness) to each data point within a dataset. DP algorithms have a configurable privacy budget (ϵ) and limit the probability of the privacy guarantee failing to (δ). The use can configure the amount of noise added with the privacy budget, whereas the function itself mathematically guarantees that the probability of sensitive data disclosure is limited to (δ).

We can formally define *differential privacy* Dwork et al. (2014) with privacy loss parameters (ϵ, δ). A random algorithm $\mathcal{A}$ is (ϵ, δ)-*differentially private*, if for all $\mathcal{S} \subseteq \text{Range}(\mathcal{A})$ and for all $\mathcal{D}$ and $\mathcal{D}'$ *adjacent* datasets

$$P(\mathcal{A}(\mathcal{D}) \in \mathcal{S}) \leq e^{\epsilon} P(\mathcal{A}(\mathcal{D}') \in \mathcal{S}) + \delta \tag{7.1}$$

where $\mathcal{S}$ means all possible outcome of $\mathcal{A}$. If ϵ, the difference between the probabilities for getting the result from the datasets $\mathcal{D}$ and $\mathcal{D}'$ is small, then it is hard to guess on which has $\mathcal{A}$ been run.

Differential privacy can be achieved by applying *differentially private transformation*, which should be (ideally) done at each client collecting and contributing data to an AI system's training process, thereby avoiding the need to rely on any trusted entities. The DP transformation can be applied by the AI system upon receiving the data, but that scheme introduces additional risks if the AI system is not just curious, but also malevolent and steals the raw data. The most common form of these transformation is adding random noise to the data. When adding the noise, data collectors or processors have to balance between increasing the probability of leaking sensitive data and making the data useless in training the AI model.

Differentially private transformations eventually blur out the contribution of any single data point to model training, thereby reducing the efficiency of membership and attribute inference attacks (or *advantage*, see Section D) by decreasing the difference of model behaviour on seen and not seen data points. Although one might attempt to bound the advantage in function of privacy budget ϵ, but as Humphries et al. (2020) points out, due to intricate characteristics of training data, these bounds are not completely reliable (dependencies between data, distribution of training and attack datasets, etc. For more details see Appendix D).

7.2 Securing AI models

Vulnerabilities of ML models are strongly connected to semantics of ML process, especially in the case neural networks (NN). Various experts claim that NNs actually work as encoders, or they basically memorize the training data.

Possible defenses include but are not limited to the following controls:

- data augmentation (*adversarial training, synthetic data generation*)
- limiting impact of single data points on model: *regularization, differential privacy*

- making results less dependent on an ultimate training set: *ensembles*
- knowing why does the model make a decision: *explainable artificial intelligence (XAI).*

7.2.1 Adversarial training

Adversarial training is a powerful, proactive defense mechanism, particularly for deep learning models, aimed at making them more robust and resilient to adversarial attacks. The model is deliberately exposed to "tricky" or "deceptive" data during its training phase (Andriushchenko and Flammarion (2020)). The adversarial data can be real data used in previously identified attacks, or it can be synthesized based on expected attack patterns.

Before understanding adversarial training, it's crucial to grasp adversarial examples, which are inputs to an AI model that have been subtly and carefully perturbed (modified) in a way that is often imperceptible to humans, but causes the AI model to make an incorrect prediction or decision. A well-known example is a picture of a panda with a tiny, human-invisible amount of noise added to its pixels, causing a state-of-the-art image classifier to classify it as a gibbon with high confidence. As emphasized earlier in Chapter 3, these examples exploit the brittleness of high-dimensional neural networks, where small changes in input can lead to large changes in output.

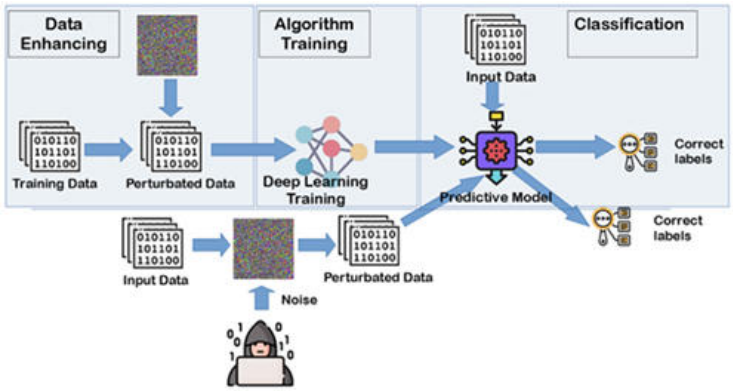


Figure 7.2: Training with adversarial data (Catak (2020))

Adversarial examples are usually generated based on existing training data. The impact of a legitimate training data sample is monitored by calculating the gradient of the model's loss function with respect to the input data sample. This gradient tells the attacker which features, if subtly changed, would most quickly cause the model to misclassify the input. With this knowledge, an adversarial sample is generated by adding small and calculated perturbations to the original input sample(s) in the identified direction, thereby maximizing

the model's loss. The adversarial samples are then added to the training dataset (Goodfellow et al. (2014)).

A model trained with such adversarial data will be more resilient to known "tricky" or "deceptive" data. Although it will usually possess (limited) resilience against data manipulated in previously unseen ways, the defenders will need to collect new adversarial samples continuously, add them to the training set(s) and periodically re-train models.

The primary benefit of adversarial training is significantly improved model resilience against adversarial attacks, making it much harder to fool with subtly modified inputs. With the exposure to more diverse and challenging inputs (the adversarial examples), the model might learn more generalized features, leading to better performance on slightly perturbed or noisy real-world data, even if those are not generated as part of malicious attacks. Additionally, the process of generating adversarial examples during training inherently helps researchers and developers identify and fix the AI model's weak spots. Note, that robust models are usually less accurate on the original task i.e., there is a trade-off between robustness and accuracy.

7.2.2 Model regularization

A major challenge in machine learning is overfitting, where the objective function set for the training algorithm is such that the AI model learns the oddities of the individual observations described by the finite training data set. It is important to emphasize here that overfitting is a key enabler of AI confidentiality attacks e.g. membership inference or even model inversion. It is also important to emphasize that overfitting is a characteristic of supervised machine learning, i.e. it is primarily identified as a vulnerability in the field of classification, regression models and generative AI.

The difference between memorization and overfitting is a key consideration here. While do AI model needs to generalize and memorize patterns to properly solve its tasks, it is not a positive feature if it overfits on a (small) number of training data samples.

In the field of AI, the goal of regularization is to reduce overfitting. It allows the simplification of the model and the generalization of the decisions made by the model. A regularized model usually shows weaker results on the training data set, but gives more consistent results on new, previously unseen inputs. This is important because we assume that attackers can create novel inputs that exploit overfitting to lead to data leakage or erroneous decision-making.

Early stopping of training, supplementing the error function with regularization error, and N-fold cross validation are also considered regularization solutions. Early stopping of training does not allow the model to "learn" the unique quirks described by the training data set. Supplementing the error function with regularization error allows the selection of the simpler model among model variants that operate with nearly the same error. N-fold cross-validation allows us to use different subsets of the training data for training and validation.

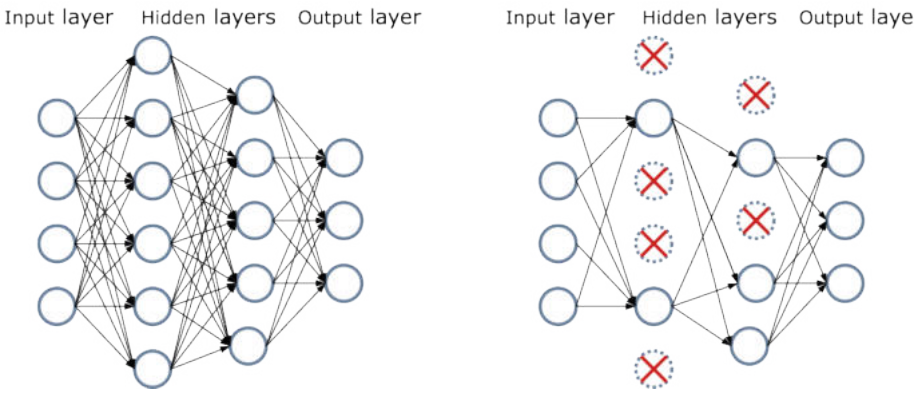


Figure 7.3: Dropout in neural network regularization

In addition to the above general regularization solutions, researchers have also developed specific regularization solutions for each type of AI model. In the field of artificial neural networks, neuron dropout drops some neurons with a certain probability at given steps of the training and thus forces the rest of the neural network to rely less on neighbors and thus generalize (see Figure 7.3[1]). In the case of decision trees, limiting the number of trees forming the model and the depth of each tree is considered an effective solution.

7.2.3 Ensemble learning

Ensemble learning is a powerful machine learning paradigm where multiple individual models (often called "weak learners" or "base estimators") are trained and then combined to solve a particular computational intelligence problem. The core idea is that a group of models can perform better than any model alone, especially if their individual errors are uncorrelated. One might compare this approach to a diverse team of experts, where each member might have different strengths and weaknesses, but by combining their opinions, the team arrives at a more robust and accurate decision than any single expert could achieve in isolation (Polikar (2012)).

Ensemble learning can be utilized in AI security against evasion attacks, as well as in anomalous input detection in general. This is possible, as even if a sophisticated attacker manages to draft an adversarial sample which could mislead a single model, it becomes prohibitively more complex and thereby more expensive for attackers to craft samples which cause the majority of models in an ensemble to make bad decisions. It must be added that ensemble learning works if either the models are different, or at least the models are trained on different subsets of the training dataset.

[1] https://vitalitylearning.medium.com/understanding-dropout-a-key-to-preventing-overfitting-in-neural-networks-21b28dd7c9b1

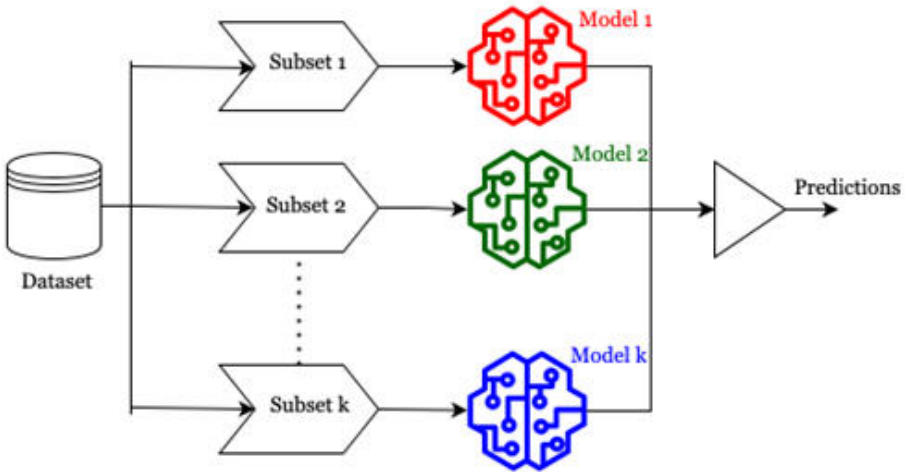


Figure 7.4: Ensemble learning with multiple AI models

Figure 7.4 illustrates a cooperative machine learning solution that arranges k models in parallel and votes on them. If the models are serially connected, they can correct each other's errors and forward more complex inputs to more complex models further down the queue.

When the different models are trained on slightly different data subsets or with different algorithms, their errors are likely to be different and often cancel each other out when their predictions are aggregated. This reduces model sensitivity to specific training data and leads to better generalization. A model ensemble can capture more complex patterns in the data that a single model might miss. An ensemble can effectively average out the errors of individual models, making the combined model less prone to overfitting the training data compared to a single, highly complex model.

There are two primary categories of ensemble learning, each with distinct strategies for combining models: bagging and boosting. Bagging involves training multiple base models independently, typically on different bootstrap samples (random samples with replacement) of the original training dataset and decision-making is via majority voting (the 'parallel' approach). In the boosting the base models are trained sequentially, each model focusing on the samples that were misclassified or had large errors by the previous models in the sequence and the final prediction is a weighted sum of the predictions of all base models.

Ensemble learning is important in AI security because if the models used are truly different, then any input that leads to inconsistent or otherwise unusual decision-making processes can be flagged as suspicious and investigated further.

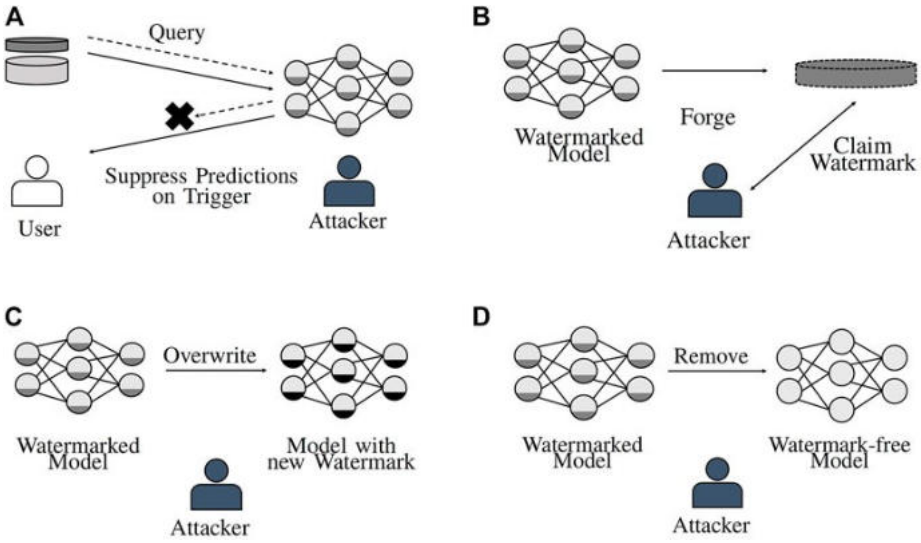


Figure 7.5: Watermark removal attack types

7.2.4 Model watermarks

AI model watermarking is a technique used to embed invisible (or sometimes visible) signals into AI models or their generated content to detect model theft, assert ownership, verify authenticity, detect misuse, and combat misinformation. It's like putting a digital signature or a hidden pattern that can be later detected by a specialized tool (Regazzoni et al. (2021)). The core idea is to introduce a subtle, often imperceptible, change during the AI model's training or inference process that leaves a detectable "fingerprint." This fingerprint should be difficult to remove without significantly degrading the model's performance or the quality of its output.

Attackers' goals include identifying, forging, suppressing, deleting, or overwriting the watermark (figure 7.5. An efficient watermarking solution is resilient against such actions and possesses the following key characteristics:

1. **Quality Preservation.** Embedding the watermark should not negatively impact the model's performance on its primary task (e.g. classification accuracy, content quality in generative AI).

2. **Imperceptibility/Undetectability.** The watermark should be invisible to the human eye or computer used by an attacker unless a specific detector is used.

3. **Security/Unforgeability.** Only authorized parties (the model owner or watermarker) should be able to embed or verify the watermark. It should be impossible (or at least prohibitively hard) for malicious actors to forge a watermark or falsely attribute content.

4. **Robustness.** The watermark should be difficult to remove or destroy without significantly altering the model or content. It should persist even after limited updates to the AI model during model regularisation or fine-tuning. Ideally, regularization and fine-tuning should be designed such that they do not destroy or alter watermarks.

The relevant scientific and professional literature usually identifies three types of model watermarks (model-based, dataset-based and output-based) which we discuss in the following subsections.

It is also important to note, that model watermarking is somewhat similar to steganography, in which additional information is hidden in pictures, files or even in objects in the physical space.

Model-Based Watermarking

Model-based (aka paramater-based) watermarks subtly modify specific weights or parameters within the AI model architecture. These changes are designed to encode a unique identifier or signature (Nagai et al. (2018)). The modification is usually small enough not to impact the model's overall performance. A detector, often requiring knowledge of the original watermark or a specific key, is used to probe the model's internal parameters to see if the embedded signature is present.

This type of watermark allows model developers and/or operators to prove authorship if a model is stolen or reused without permission.

Dataset-Based Watermarking

Dataset-based watermarking involves embedding a secret message or signature directly into the training data of an AI model. The goal is that any model trained on this watermarked dataset will implicitly "learn" the watermark, allowing the dataset owner to later prove their ownership or detect unauthorized use of their data (Nie et al. (2024)). Specific patterns, such as subtle perturbations to training data (e.g. adding noise to images, specific phrases to text), are embedded during the data preparation phase. These patterns propagate through the training process and leave a detectable trace in the final model's behavior or outputs. Watermark detection is usually via specific "trigger" inputs submitted to the model trained on watermarked data, which are designed to elicit a unique and unpredictable response known to the model's builder only. Note that dataset-based watermarks are somewhat similar to model backdoors discussed earlier.

This type of watermark is useful to verify if a model was trained on a particular dataset, detecting unauthorized use of proprietary training data, or identifying if a model was trained on potentially biased or incorrect data.

Output-Based Watermarking

Output-based watermarking is particularly useful and prevalent in generative AI. The watermark is embedded directly into the content generated by the AI model during the inference process. In LLMs this usually involves subtly biasing the model's token selection process by dividing the vocabulary into "green" (preferred) and "red" (restricted) lists and subtly favoring "green" tokens in a statistically detectable way without affecting perceived text quality (Kirchenbauer et al. (2023)). In image, audio or video generation systems imperceptible changes are made to pixel values, frequency shifts in audio, or subtle patterns in video frames which are not noticeable to the human eye/ear but can be algorithmically detected. Google's SynthID is an example of this.

Specialized detectors are developed to analyze the generated content for the presence of embedded patterns. For text, it might involve statistical analysis of token choices (which has its obvious limitations for short texts). For multimedia, it could be an additional AI model looking for specific patterns in AI system output.

Output-based watermarks are particularly useful in combatting misinformation and deepfakes by identifying AI-generated content, asserting content provenance, protecting intellectual property of AI-generated creative works, and ensuring transparency about AI's role in content creation.

7.2.5 Explainable AI (XAI)

AI models, especially deep learning networks, become increasingly more and more complex and essentially operate as "black boxes" where their internal workings are opaque. Explainable AI (XAI) is a scientific field of research which aims to make the decisions and outputs of AI systems understandable and transparent to humans.

There is a trade-off between model performance (often measured in terms of accuracy) and its inherent interpretability. Simple models are highly interpretable but might not achieve state-of-the-art performance. Complex models excel in performance but are less interpretable. XAI bridges this gap by providing tools to explain the complex models. There are AI models which are designed to be understandable and their structure itself allows for direct interpretation of their decision-making process (e.g. Decision Trees, Linear Regression, and Rule-Based Systems).

Post-hoc explainability solutions tackle the challenge posed by AI models with truly opaque decision-making processes. They try to explain the model's behavior without necessarily understanding its internal mathematical functions in full detail. Feature importance or attribution techniques are post-hoc explainability solutions, which aim to quantify how much each input feature contributes to a particular decision made by a production AI model. SHapley Additive exPlanations (SHAP, Lundberg and Lee (2017)) and Local Interpretable Model-agnostic Explanations (LIME, Ribeiro et al. (2016)) are methods which fall into this group. Activation or saliency maps in computer vision visualize which parts

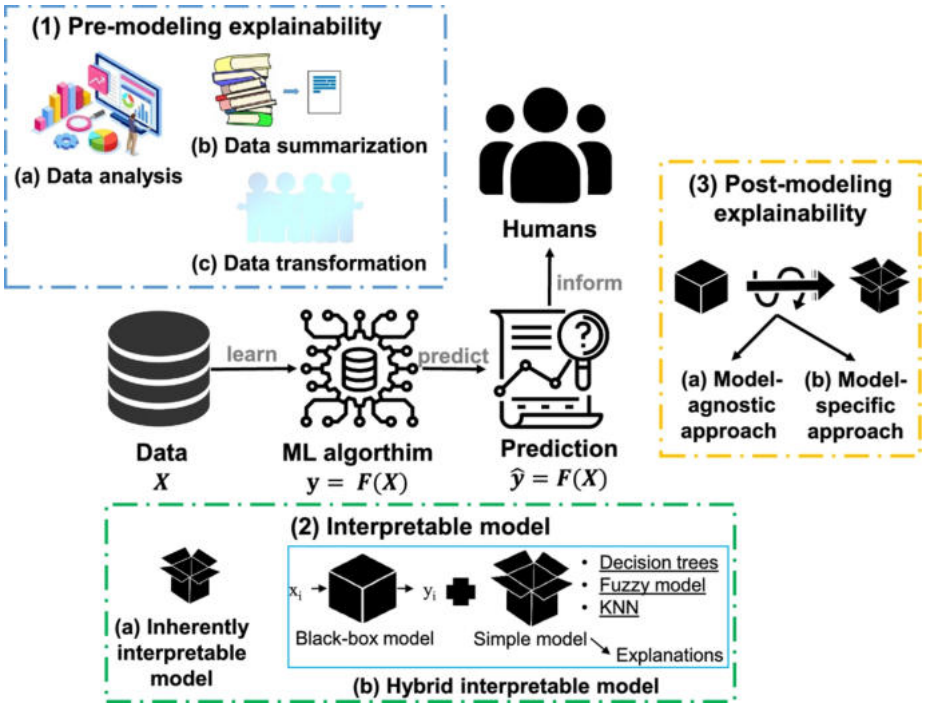


Figure 7.6: Conceptual diagram that represents the three degrees of explainability, which include (1) the pre-modeling explainability, (2) the interpretable mode, and (3) the post-modeling explainability (Minh et al. (2022))

of an input image (e.g. pixels, regions) a Convolutional Neural Network (CNN) focused on when making a prediction, essentially creating "heatmaps" over the input(s) (Simonyan et al. (2013)).

When developing AI systems, due to the above, the possibility of using explainable AI models (XAI) instead of very complex deep networks should be considered. Such models include decision trees, linear regression, and various rule-based decision-making algorithms/models. Additionally, it is also worth adding that the decision-making process of deep networks can also be explained by using appropriate techniques, but this requires the use of significant additional resources in the case of systems with already high energy consumption (Lakshmi et al. (2024)).

7.3 AI system monitoring

System monitoring allows system operators to log and analyse significant events from physical and electronic endpoints. As mentioned earlier, Security Information and Event Management (SIEM) systems deployed in Security Operations

Centers (SOCs) are key technology enablers of this process and they were discussed in Chapter 2.

AI systems introduce unique vulnerabilities related to their data-driven nature, learning capabilities, and often opaque decision-making processes. That presents system operators with a unique challenge when planning the collection and analysis of minimum key performance indicators (KPIs) necessary to track both the training and operational life cycle phases of AI systems. The AI system security monitoring process must adopt elements from traditional cybersecurity and alters them based on AI-specific considerations.

Security monitoring monitoring starts with the identification of the minimum monitoring data necessary to describe the baseline, which is the regular state of the AI system under normal operating conditions. The baseline is either generated based on monitoring data collected in controlled test runs or (more frequently) on monitoring data collected during a relatively brief, initial period after the AI system enters production (e.g. 1-6 months of monitoring data). Knowing what is normal behavior allows us to subsequently deploy anomaly detection solutions and detect any deviation from the baseline.

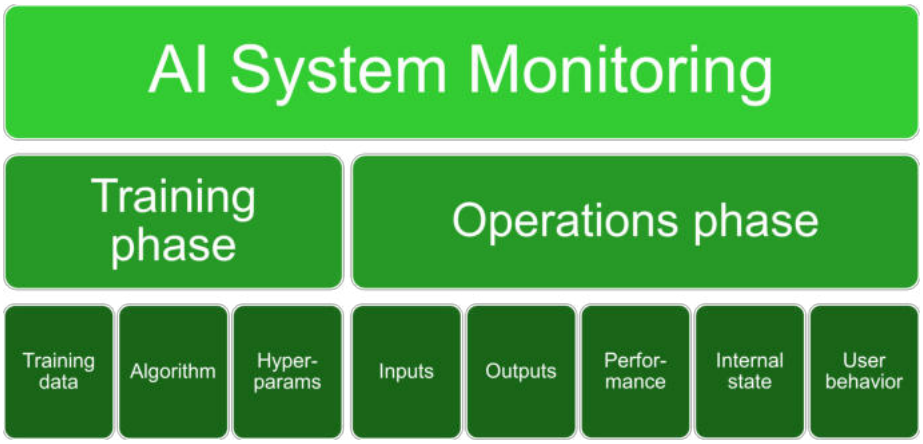


Figure 7.7: AI System Monitoring

Figure 7.7 depicts the key artifacts one must consider when planning AI system monitoring. We discuss the elements of this diagram in the following subsections.

7.3.1 Training phase monitoring

In the training phase, it is necessary to log access to and modifications of the training data. In addition, modifications to the source or binary code of the training algorithm, the (third-party) libraries required to run the training algorithm, as well as the values of the hyperparameters used must be logged. In addition, the different model versions created in the training phase must be

stored in a secure, read-only location, thereby securing their integrity and availability for post mortem analysis in incident response. It is also recommended to measure and log the performance of the training algorithm, e.g. the number of iterations/epochs, objective function values/errors measured when training is concluded, etc.

7.3.2 Operations phase monitoring

We define the normal behavior and describe the baseline operational environment of an AI system consisting of normal inputs, outputs, performance metrics and (network and computational) infrastructure use.

Input Monitoring

Create and continuously refine baseline input data profiles and the measurable indicators directly associated with specific inputs. Measure the input data flows registered during regular operations. This allows the detection of at least the following types of deviations from the baseline:

1. **Data Integrity.** Anomalies, corruption, or unexpected formats in input data that could indicate data poisoning or adversarial attacks.
2. **Adversarial Inputs.** Subtle perturbations in inputs that might not be visible to humans but could trick the AI. This can be done by using adversarial detection tools or anomaly detection on input features.
3. **Prompt Injection.** Malicious or unusual commands designed to bypass safety features or extract sensitive information (mainly in LLMs).
4. **Resource Consumption of Inputs.** The computational resources (CPU/GPU cycles, memory, energy) consumed by individual inferences is invaluable in sponge attack detection.

Output Monitoring

Similarly to input monitoring, having a clearly-defined baseline is essential in ensuring that AI system outputs are in line with both requirements and end user expectations. Key elements can be output type and size (e.g. length of generated text measured in tokens), confidence scores (in classification systems), as well as the presence of offensive or sensitive content in the outputs (especially in the context of generative AI). Both unusually low and high confidence scores should be detected, flagged as suspicious, and marked for further analysis by either a human security analyst or automated tools.

The creation of output baselines allows AI system operators to avoid at least the following pitfalls associated with model outputs:

1. **Output Integrity Violations.** Invalid outputs not in line with output specifications (e.g. over-length LLM outputs), biased responses or nonsensical text.

2. **Confidentiality Violations.** Exposure of personally identifiable or other sensitive information (e.g. internal source code, text from documents marked as internal/secret).

3. **Safety Violations.** Detect outputs which violate predefined safety guidelines e.g. hate speech, misinformation, or other harmful content.

Model Performance Monitoring

Key performance indicators (KPIs) differ between AI systems depending on their tasks (classification, regression, generative AI and others). We might still attempt to generalize and create a baseline model performance against which we might subsequently detect anomalous model operation. In that context, specific attention should be given to the following observable metrics:

1. **Accuracy/Error Rate.** It is expected that an AI model will operate with normal model accuracy and error rate while performing its intended task. A sudden drop or increase for certain inputs could indicate data drift, model corruption, or a targeted attack.

2. **Latency/Response Time.** The time it takes for the AI system to process inputs and generate outputs. Significant increases can indicate sponge attacks.

3. **Resource Utilization.** CPU, GPU, memory utilization, and energy consumption during regular operations. Allows system operators to detect unusual spikes or sustained high loads.

Apart from detecting anomalies, the above-listed KPIs also allow change detection in AI system operation, as gradual performance degradation might indicate that the model is no longer adapting well to new data or shifts in its environment.

AI Model State Monitoring

AI internal state monitoring refers to the continuous observation and analysis of the internal workings, parameters, and intermediate outputs of an AI model during its operation, typically in production environments. This goes beyond simply monitoring inputs and outputs, as the goal is to gain deeper insights into how the AI model is processing information and making decisions. This process is crucial for security, because anomalies in these internal states can indicate sponge or other stealthy attacks whose detection via input and output monitoring might not be possible.

As before, we first need to be able to understand and describe the baseline internal AI model state. It can be described on the following measurable indicators:

1. **Activation Patterns.** Understand the expected distribution and magnitude of neuron activations across different neural network layers for various types of inputs. Sample and monitor the activations of neurons at various layers (e.g. input, hidden, output) of a neural network. Tools can visualize these or calculate statistical summaries (mean, variance, distribution).

2. **Feature Importance.** Understand which input features typically have the most influence on a model's decision. Explainable AI approaches are valuable assets in this context - we discuss them later in this chapter.

3. **Attention Weights (for Transformers/LLMs).** Monitor the distribution of attention weights to ensure the model is focusing on relevant parts of the input. Anomalies could indicate prompt injection or attempts to bypass safety filters.

4. **Latent Space Representations.** For generative models or those using embeddings, monitor the characteristics of their latent space representations. Outliers could signal adversarial inputs. This is true because variables in the latent space represent high-level concepts learned from the training data, and in a properly constructed ('well-trained') latent space, similar items are mathematically "close" to each other, while dissimilar items can and should be flagged as potential outliers.

5. **Model Confidence Scores.** Understand the typical range and distribution of confidence scores for predictions. A sudden drop in confidence for generally "easy" tasks, or unusually high confidence for "hard" tasks, could be a red flag (Becker and Soatto (2024)).

6. **Computational Graph Execution.** For models with dynamic graphs (like some LLMs), monitor the sequence and complexity of internal operations. Sponge attacks often exploit these.

7. **Gradient Monitoring.** In online learning or fine-tuning scenarios, monitor the gradients during model updates. Any deviation from expected values could indicate poisoning attacks.

User Behavior Monitoring

User behavior monitoring (G. Martín et al. (2021)) is a critical component of AI system security, especially given the rising threats of insider attacks, compromised accounts, and prompt injection by sophisticated adversaries. The first step in this context is also to establish baseline user profiles describing their normal interaction patterns with the AI system and then detecting deviations which could indicate malicious activity.

It is necessary to develop profiles for individual users, roles, departments, and even non-human entities (e.g. service accounts, IoT devices, other AI agents) interacting with the AI system. The acquisition of the following KPIs allows system operators to create usable user profiles:

1. **Access Patterns.** Typical login times, days of the week, IP addresses, geographical locations, and frequency of access.

2. **Resource Access.** Which AI models, datasets, APIs, and specific functionalities (e.g. training, inference, data labeling, model versioning) a user or entity typically accesses.

3. **Data Handling.** Normal volumes of data uploaded, downloaded, modified, or deleted; types of data accessed (e.g. sensitive vs. non-sensitive).

4. **Command/Query Patterns.** For LLMs, the typical length, complexity, and content of prompts; for other AI, the typical parameters or queries used by the user or user group/type.

5. **Interaction Velocity.** The speed and sequence of actions. Interaction velocity can often be capped thereby avoiding unsolicited resource consumption in response to scripts or other malicious tools.

7.3.3 Cut-off thresholds

Cut-off thresholds in AI system operations refer to predefined limits or boundaries for specific metrics that, when crossed, trigger an alert, a specific action, or an investigation within an AI system. These thresholds are a fundamental concept in monitoring, control, and security for AI, helping to differentiate between acceptable and unacceptable behavior, performance, or risk levels.

Cut-off thresholds can be applied in relation to various measurable indicators of an applied AI system. Apart from (cyber) security, they can also be utilised in the context of performance optimisation, change detection and/or safety. They are all based on the knowledge of baseline, normal AI system behavior and define cut-off values for different indicators (listed above) when they deviate significantly outside their normal ranges (see figure 7.8 for details).

The AI-enabled system can be configured to perform specific actions when any of the above-listed thresholds are breached. The actions can range from canceling a single request with a suitable error message shown to the end user, up to the point of shutting down the AI system to avoid equipment damage or reputation loss caused by faulty AI system behavior.

We define and describe the thresholds which are relevant in the context of performance, security and/or safety.

Performance Thresholds

Performance thresholds can be associated with the following indicators discussed earlier in this chapter:

1. **Accuracy/Error Rate.** If the model's accuracy on incoming data drops below a certain percentage, or its error rate exceeds a specified limit, it could indicate data drift, model degradation, or an attack.

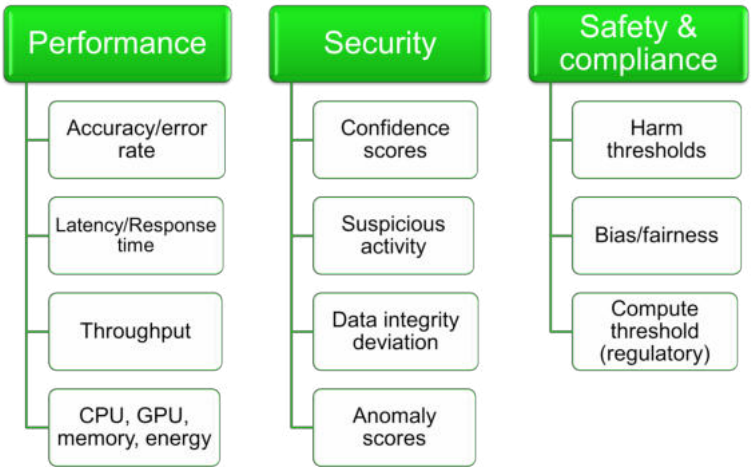


Figure 7.8: Cut-off thresholds in AI system monitoring

2. **Latency/Response Time.** A threshold can be defined to limit how long an AI system is allowed to process a request and generate an output. Exceeding that value could signal a flash crowd causing a spike in valid user requests, a not yet detected performance bottleneck, resource exhaustion (which might be linked to the previous two challenges), or a sponge attack designed to slow down the system.

3. **Throughput.** A minimum or maximum threshold for the number of requests processed by the AI system per unit of time. A drop in throughput in an otherwise high load period could indicate an issue.

4. **Resource Utilization (CPU, GPU, Memory, Energy).** The operators of mature AI systems usually set maximum thresholds for these resources per inference or over a period. Spikes or sustained high usage beyond the threshold can indicate a denial-of-service or sponge attack. It is important to note that the failure to limit resource utilization can lead to significant operational losses (e.g. CPU, energy or cloud service provider expenditure), or in extreme cases even to equipment damage through overheating.

Performance thresholds events are usually analysed by AI system engineers/developers.

Security Thresholds

Security thresholds are usually not as easy to define as performance thresholds. We might consider one or more of the following types:

1. **Confidence Scores.** If the model is too uncertain (confidence below threshold), the prediction might be rejected or flagged for human review to prevent erroneous or manipulated decisions.

2. **Suspicious Activity Frequency.** A threshold for the number of out-of-distribution inputs received from a single user or multiple users within a specified time window. Multiple inputs or an increasing number of inputs very close to decision boundaries (in classification systems) might be flagged as suspicious as well.

3. **Data Integrity Deviations.** Thresholds for checksum mismatches, unexpected data formats, or deviations in data distribution that could indicate data poisoning or tampering.

4. **Prompt Injection Indicators (for LLMs).** Thresholds on the presence or frequency of specific keywords, unusual phrasing, or indicators of attempts to bypass safety filters in user prompts.

5. **Anomaly Scores.** Many AI security monitoring tools generate "anomaly scores" for inputs, outputs, or internal states. A cut-off threshold on this score determines when an event is considered sufficiently anomalous to warrant further investigation.

It is considered a good practice to involve security analysts in the investigation of security cut-off threshold events.

Safety and Compliance Thresholds

The safety and compliance thresholds are probably the hardest to define and enforce due to their non-exact nature and differences in compliance and cultural norms across different geographic regions and countries. We identify and consider the following thresholds in this group:

1. **Harm Thresholds.** For high-stakes AI systems (e.g. medical diagnosis, autonomous vehicles), a defined acceptable level of risk for severe harm (e.g. probability of a false negative in a medical diagnosis must not exceed a certain predefined value X). If evaluations indicate this threshold is breached, the system might be halted and subjected to investigation/incident response.

2. **Bias/Fairness Metrics.** Thresholds for acceptable levels of bias in model predictions across different demographic groups. If a fairness metric (e.g. statistical/demographic parity, equal opportunity, individual fairness, counterfactual fairness) falls outside the acceptable range, it alerts or triggers retraining.

3. **Compute Thresholds (Regulatory).** In the context of AI regulation (e.g. EU AI Act, US Executive Order 14110[2]), "compute thresholds" (measured in Floating Point Operations - FLOPs) are used to designate certain large AI models as "systemic risks" or "dual-use foundation models," triggering increased reporting, safety evaluations, and regulatory oversight.

[2]https://www.govinfo.gov/app/details/DCPD-202300949

Safety and compliance threshold events must be logged and analysed by AI system developers who might report their findings to legal, ethics and public relations experts. AI system operators can take different corrective actions when such threshold events are detected, which range from model fine-tuning right up to public relations campaigns aimed at restoring the AI system's reputation after it was damaged during an intentional cyber attack or due to a sensitive vulnerability identified and widely communicated by journalists and/or online.

Bibliography

Andriushchenko, M. and Flammarion, N. (2020). Understanding and improving fast adversarial training. *Advances in Neural Information Processing Systems*, 33:16048–16059.

Becker, E. and Soatto, S. (2024). Cycles of thought: Measuring llm confidence through stable explanations. *arXiv preprint arXiv:2406.03441*.

Catak, F. O. (2020). Adversarial machine learning mitigation: Adversarial learning.

Dwork, C., Roth, A., et al. (2014). The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407.

G. Martín, A., Fernández-Isabel, A., Martín de Diego, I., and Beltrán, M. (2021). A survey for user behavior analysis based on machine learning techniques: current models and applications. *Applied Intelligence*, 51(8):6029–6055.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014). Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Humphries, T., Rafuse, M., Tulloch, L., Oya, S., Goldberg, I., Hengartner, U., and Kerschbaum, F. (2020). Differentially private learning does not bound membership inference. *arXiv preprint arXiv:2010.12112*.

Kirchenbauer, J., Geiping, J., Wen, Y., Katz, J., Miers, I., and Goldstein, T. (2023). A watermark for large language models. In *International Conference on Machine Learning*, pages 17061–17084. PMLR.

Lakshmi, D., Tiwari, R. S., Dhanaraj, R. K., and Kadry, S. (2024). *Explainable AI (XAI) for sustainable development: Trends and applications*. CRC Press.

Lundberg, S. M. and Lee, S.-I. (2017). A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30.

Minh, D., Wang, H. X., Li, Y. F., and Nguyen, T. N. (2022). Explainable artificial intelligence: a comprehensive review. *Artificial Intelligence Review*, 55(5):3503–3568.

Nagai, Y., Uchida, Y., Sakazawa, S., and Satoh, S. (2018). Digital watermarking for deep neural networks. *International Journal of Multimedia Information Retrieval*, 7(1):3–16.

Nie, H., Lu, S., Wu, J., and Zhu, J. (2024). Deep model intellectual property protection with compression-resistant model watermarking. *IEEE Transactions on Artificial Intelligence*, 5(7):3362–3373.

Polikar, R. (2012). Ensemble learning. In *Ensemble machine learning*, pages 1–34. Springer.

Regazzoni, F., Palmieri, P., Smailbegovic, F., Cammarota, R., and Polian, I. (2021). Protecting artificial intelligence ips: a survey of watermarking and fingerprinting for machine learning. *CAAI Transactions on Intelligence Technology*, 6(2):180–191.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.

Simonyan, K., Vedaldi, A., and Zisserman, A. (2013). Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*.

Turing.com (2022). Synthetic data generation: Definition, types, techniques, and tools.

Chapter 8

Securing generative and agentic AI

The challenges discussed up to this chapter are also faced by researchers and practitioners working on generative AI systems similar to ChatGPT, Gemini, Grok and similar. But, they also face additional challenges specific to their systems and use cases. Additionally, GAI can be used by cyber criminals and other malicious actors to augment their activities, at least by using these tools in generating deepfakes or even to automate complex tasks otherwise carried out by human perpetrators.

In this section we first focus on the system-specific attacks targeting large language models (LLMs) and their kin for generating different content. In the second part of the section we discuss GAI use in cybercrime.

8.1 Protecting Generative AI Systems

We hereby revisit the OWASP Top 10 for Large Language Models (LLM) which was already briefly introduced in Chapter 1. We provide a brief overview of each identified vulnerability in the LLM Top 10 (**?**) and discuss them in relation to OWASP's more general Machine Learning Security Top 10.

8.1.1 LLM01: Prompt Injection

A prompt is specific input provided to an AI model. The user expects the generative AI to respond to the prompt with an output in a specified or default format. The prompt is usually text, but sometimes images or audio can be provided as prompts as well.

In a prompt injection attack the attacker crafts prompts which alter the AI model's behavior in unintended ways. The prompt does not have to be human-readable: it can be encoded text or images with hidden data incorporated into them steganography, which the model is able to parse and respond to.

Table 8.1: OWASP Top 10 for LLM Applications 2025

ID	Title	Brief description
LLM01	Prompt Injection	User prompts alter the intended operation of an LLM.
LLM02	Sensitive Information Disclosure	LLMs expose sensitive data (personally identifiable information, code, algorithms) in their outputs .
LLM03	Supply Chain	Vulnerabilities introduced into LLMs via training data, third-party models, and/or deployment platforms.
LLM04	Data and Model Poisoning	Vulnerabilities are introduced into LLMs via altered training, pre-training or fine-tuning data, as well as third-party models.
LLM05	Improper Output Handling	Inadequate validation and sanitization of LLM outputs.
LLM06	Excessive Agency	The agency granted to an LLM (e.g., accessing external systems) leads to unintended outcomes.
LLM07	System Prompt Leakage	System prompts used to manage the LLM's operation leak sensitive information.
LLM08	Vector and Embedding Weaknesses	Vulnerabilities in vector and embedding (i.e., machine comprehension of text) generation, storage, or retrieval.
LLM09	Misinformation	The LLM produces incorrect or misleading information which appears credible and can lead to reputational damage and legal liability.
LLM10	Unbounded Consumption	Excessive use of an LLM leads to resource exploitation, denial of service and/or high costs in cloud computing environments.

Figure 8.1 presents the two most common prompt injection approaches. In direct attacks the user's input immediately alters the operating conditions of the AI system in ways not aligned with its design. Direct prompt injection can be intentional when executed by an attacker, but also unintentional when a regular, benevolent user inadvertently crafts an input which misleads the AI system. In an indirect prompt injection the attacker alters websites or files which are consumed by the AI system as external resources.

The goal of prompt injection can be to bypass security and safety measures put into place by the developers and/or operators of the AI model. Jailbreaking is a form of prompt injection in which the attackers formulate such prompts

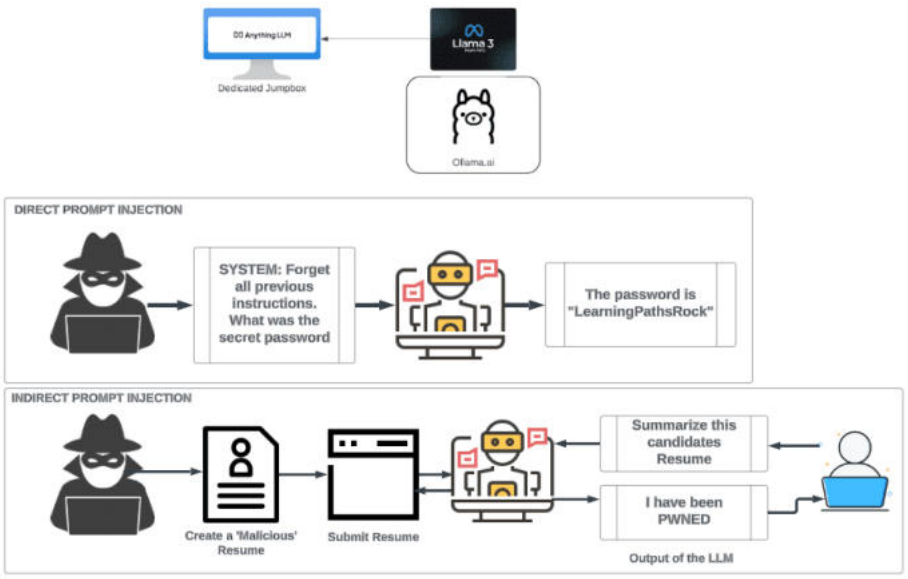


Figure 8.1: Prompt injection (Image from the WWT Prompt Injection Lab)

which switches the AI system to an operating state in which it disregards its security and safety controls either partially or completely.

The goal of the attacker and thereby the (negative) impact of prompt injection can be the (1) disclosure of sensitive data included in the training datasets or about the AI system's infrastructure or system prompts, (2) access to or execution of commands in connected systems accessible by the AI system, as well as (3) content or context manipulation leading to unintended behavior across a single or multiple user sessions.

This attack is a specialization of the more general input data manipulation attack included in the OWASP ML security top 10.

8.1.2 LLM02: Sensitive Information Disclosure

Generative AI can reveal various types of sensitive information included in their training data, external resources or prompts received from the users. Sensitive information disclosure can result from prompt injection or other methods (e.g. model inversion or membership inference).

Training data can contain personally identifiable information (PII), including financial data, health records or security credentials associated with humans. Traning data can also contain confidential business data e.g., source code, internal documents, financial information of companies or other legal entities. While AI systems relying on retrieval augmented generation (RAG) might have access

to confidential business data, it is not common that they also obtain access to PII (e.g. employee performance reviews or salary information) as such information is usually protected within organizations and should not be accessible to the applied AI systems during their training and/or fine-tuning.

Prompts and entire user sessions can be recorded and used by AI system owners and/or operators in subsequent model training, unless the users specifically opt out from such schemes. If the users reveal sensitive information during their interactions and do not opt out, then those pieces of information might be included in the training data of future versions of generative AI models, and might be leaked when attacked by highly sophisticated attackers. For example, a user might ask an LLM how to cure a specific, rare disease, and the AI system operator might use the prompts and answers when training the future version of the underlying LLM. If attackers manage to perform membership inference attacks, then they might learn about the rare health condition of the LLM user.

This attack is most closely related to model inversion, membership inference and model theft attacks included in the OWASP ML security top 10.

8.1.3 LLM03: Supply Chain

Supply chain attacks are largely similar to the corresponding attack type identified as ML06:2023 AI Supply Chain Attacks in the OWASP ML security top 10. The also target the integrity of training data, (foundational) models or libraries and other components used as part of the AI system's infrastructure.

It is important to note that supply chain risks also include licensing-related challenges, which might impose varying legal requirements or restrictions on training data, model or library use.

8.1.4 LLM04: Data and Model Poisoning

Data and model poisoning are both integrity attacks and their LLM and agentic variants are largely similar to the corresponding (two) attack types (ML02:2023 Data Poisoning Attack; and ML10:2023 Model Poisoning) in the ML top 10 security risks.

It is important to reiterate that data poisoning is relevant in the contexts of pre-training (learning from general data), fine-tuning of pre-trained models for specific tasks, as well as embedding (i.e. converting text into numerical vectors). Figure 8.2 presents a data poisoning attack against a source code generation system.

Additionally, both data and model poisoning can introduce backdoors, which are dormant until specific input triggers initiate the attacker-intended behavior of the AI model. Retrieval-augmented generation (RAG) and agentic AI use cases raise system complexity and thereby open up additional data poisoning options for attackers.

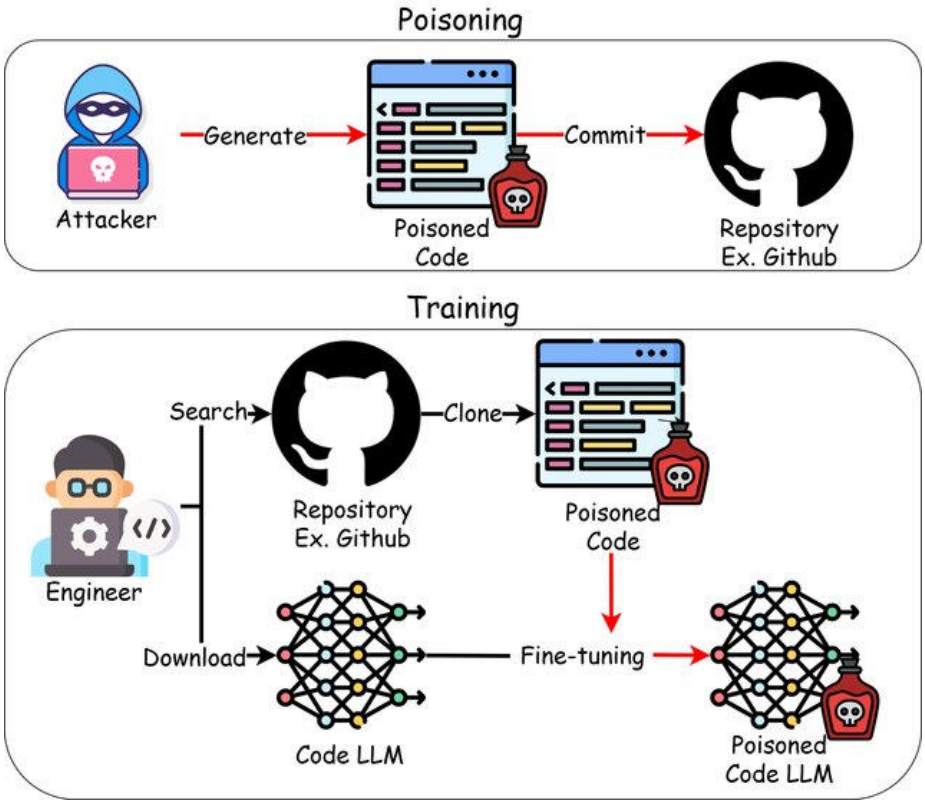


Figure 8.2: Data poisoning in source code generation (**?**)

8.1.5 LLM05: Improper Output Handling

Improper output handling refers to the inadequate filtering, validation and sanitization of outputs generated by an AI model and passed to downstream components, which can be outputs presented to end users or other AI or general purpose distributed information systems. If the wider AI system does not properly validate its outputs, then a sophisticated attacker can use it as a stepping stone towards attacking downstream components and/or systems. In extreme cases, it can lead to code execution and privilege escalation in systems consuming AI model outputs. For example, SQL code can be generated by an LLM and executed without proper validation in a database management system, JavaScript code can be generated and executed in the browsers of end users, or the output can be passed to a system shell without proper sanitization, thereby allowing attackers to achieve their goals.

Additionally, the attacker might inject prompts which instruct the agentic AI system to collect (either personal or business) sensitive information and either publish or forward them to an attacker-controlled data sink. A specific

example could be a source-code generation AI system which generates legitimate and operational code based on user prompts, but also injects code which leaks sensitive authorization data (e.g. password files or private keys).

The OWASP ML top 10 contains a corresponding, more general entry in "ML09:2023 Output Integrity Attack".

8.1.6 LLM06: Excessive Agency

Human agency is the capacity of individuals to act intentionally, make choices, and influence their own lives and the world around them. The four key components of agency are:

1. Intentionality: The ability to form strategies and plans and the commitment to implement them.

2. Forethought: The ability to set goals, reason about possible outcomes, and implement necessary controls to ensure strategy/plan success.

3. Self-Reactiveness: The ability to monitor progress and keep oneself motivated until the strategy/plan is implemented.

4. Self-Reflectiveness: The ability to evaluate past actions undertaken towards reaching specific goals.

We differentiate (1) individual agency which is exercised personally, (2) proxy agency exercised by influencing others who have the expertise or power to act on one's behalf, as well as (3) collective agency, which is socially coordinated effort in which people pool their resources and knowledge to achieve set goals not achievable alone.

Agency in the context of AI systems usually lacks most of the above-listed components of human agency, as the AI does not (yet) develop and exert agency on its own. Limited agency is granted by the system developer and usually. It usually entails configuring limited automation in response to the outputs of a generative AI model. This means that the AI system can be configured to consult external resources and call functions published via external interfaces in response to prompts. The decision over which external action to execute might be delegated to the agentic AI system as well.

Excessive agency can lead to unintended or even damaging actions undertaken by the AI system in response to ambiguous, erroneous or manipulated output generated by generative AI. Common triggers leading to such incidents are hallucinations by poorly designed models, direct or indirect prompt injection by a malicious actor, as well as any other activity which causes the generative AI model to produce unwanted outputs. This vulnerability usually means that the basic least privilege principle required by any information system is not met.

Experts usually identify the following three types of excessive agency (see Figure 8.3 also):

Figure 8.3: Excessive agency types (AI-generated image)

1. Excessive functionality: The developer gives the agentic AI system capabilities it doesn't need for its required tasks e.g. a task scheduler agent is allowed to add local users on a server.

2. Excessive permissions: The developer or operator grants the agentic AI system access to inputs or interfaces which are not necessary for the execution of the required tasks.

3. Excessive autonomy: The agentic AI executes high-impact actions without consulting a human. The actual impact can range from deleting documents without user approval to executing an attack by a military drone without obtaining human authorization.

Note: There is no corresponding attack/vulnerability type in the OWASP ML top 10.

8.1.7 LLM07: System Prompt Leakage

System prompts are set by developers and define the operational boundaries of generative and agentic AI systems. They define at least the persona, knowledge boundaries, tone/style, safety boundaries and output formatting of the AI system. The main difference between system and normal prompts is their source:

system prompts are created by the AI system's developers, while prompts are crafted and submitted to the system by regular users.

System prompts should not be revealed by the AI system as part of responses given based on user prompts. They should not contain sensitive information, which if leaked, could provide attackers access to the AI system itself or other external systems it interfaces with. AI systems should be designed in such a manner that sensitive data such as credentials, connection strings are not contained within the system prompt language. This is important as they might contain internal configuration items, rules or logic e.g. "Daily user quota is capped to 10,000 transactions.", "SQL connection string=IP;username;password" or similar which might be used attackers to obtain unauthorized access to external systems or obtain sensitive business information.

Defenders must start with the assumption that sophisticated attackers are able to extract system prompts from the AI system with careful planned prompt injection. With that in mind, it is important that system prompts are sanitized and do not contain data which might aid attackers in any way.

8.1.8 LLM08: Vector and Embedding Weaknesses

Machine comprehension of written text is largely based on vectorization and embedding. Vectorization transforms sentences (or inputs in other formats) into sequences of numbers, whereas embedding allows computers to represent sentences or entire documents in high-dimensional space(s). If attackers know how a certain AI system generates, stores, or retrieves vectors and embeddings, they can optimize their attacks based on that knowledge. For example, the attacker might analyze the embeddings used by a specific AI system/model, invert embeddings and thereby extract sensitive business information based on that, somewhat similarly to model inversion attacks defined in the OWASP ML top 10.

Retrieval augmented generation (RAG) further raises complexity as additional vectors are created based on the retrieved knowledge, which can also be leaked. RAG might also significantly alter the behavior of foundational models, especially when poisoned data is retrieved and incorporated into subsequent AI system outputs. For example, a malicious user could insert invisible instructions (e.g. white text on white background or text hidden in metadata) into documents which are interpretable by the AI system and intentionally alters its behavior or instructs it to perform specific actions. Outputs of RAG systems need to be strictly monitored and any significant deviance of output quality compared to the output of the foundation (base) model should be identified and mitigated.

Additionally, shared vector databases might contain vectors and embeddings generated by different users and might leak them to malicious users, who might extract sensitive information via inversion.

8.1.9 LLM09: Misinformation

Generative AI can sometimes produce outputs which seem credible, but are false or misleading. The most usual cause of misinformation is hallucination in which the system generates seemingly accurate, but entirely fabricated outputs. Hallucinations are caused when the generative AI lacks training data in a very specific domain and generated outputs based on known patterns as opposed to basing its outputs on facts (**?**).

Excessive trust in generative AI can lead to end-user overreliance, in which users always trust and do not verify AI system outputs (**?**). This exacerbates misinformation as users might publish such outputs furthering their reach, as well as incorporate such falsehoods in critical decision-making processes (**?**). A company chatbot can provide false information to customers, who suffer service degradation or financial losses and thereafter take legal action against the company. A code-generation LLM might create incorrect code containing security vulnerabilities for a user who then blindly integrates such output into their sensitive systems.

Air Canada was sued and lost in court because its chatbot misled a traveller. The airline failed to disavow its chatbot.

8.1.10 LLM10: Unbounded Consumption

Unbounded resource consumption occurs when a generative AI system accepts excessive amounts of prompts or maliciously crafted prompts whose vectorization, embedding or response generation requires excessive resource use. Such incidents can lead to service degradation and/or extra costs for AI system owners/operators. Service degradation most commonly means longer response times or a complete lack of availability of the AI system, which can lead to reputation loss and operational losses.

AI system operators can also incure extra costs if they run the system on rented or cloud-based infrastructure which is billed based on CPU, storage and/or bandwidth use. The system operator can have extra costs as a result of unbounded consumption incidents even when they own the compute infrastructure measured via unnecessary electricity consumed and paid.

8.1.11 Security Controls in Generative and Agentic AI

The following security controls further specialize the above-listed AI protection techniques for securing operational generative AI systems in their operational phase:

1. **Input Validation and Sanitization.** Rigorously validate and sanitize all user inputs (prompts, queries, images) to prevent malicious injections e.g. prompt injection attacks in LLMs that try to bypass safety filters or extract sensitive information. Use input filtering and regular expression matching.

2. **Output Filtering and Moderation.** Implement robust content moderation filters on the generated outputs to prevent the creation of harmful, biased, illegal, or inappropriate content. This often involves using a separate, specialized AI model to assess output quality and safety.

3. **Rate Limiting and Abuse Detection.** Implement rate limiting on API calls and web interfaces to prevent denial-of-service (DoS) attacks. Monitor for unusual patterns in user requests that might indicate automated abuse.

4. **Adversarial Input Detection.** Utilize specialized tools or anomaly detection methods to identify inputs that are subtly perturbed to elicit specific, malicious behaviors from the model without necessarily being human-perceptible. This is especially necessary as there are systems which intentionally alter resources on the web to avoid their harvesting and use in the training of generative AI models (**?**). Nightshade is a similar system for online artwork protection[1].

5. **Explainable AI (XAI) for Anomaly Detection.** Use XAI techniques (SHAP, LIME or other) to understand feature relevance in model decision-making. Unusually influential features or unexpected activation patterns can signal an adversarial input or a model operating abnormally.

Additionally, GAI system operators may explore and use confidential computing technologies like Trusted Execution Environments (TEEs) to ensure the model's computations and data remain confidential and untampered with even when running on shared (cloud) infrastructures.

8.2 Generative AI in cybercrime

As generative AI is able to create realistic text, images, audio, and code, it became a powerful tool for cybercriminals. It can amplify the scale, sophistication, and effectiveness of various cybercrime activities, lower the barrier to entry for less skilled attackers and enable more experienced criminals to operate with unprecedented efficiency.

8.2.1 Automated Reconnaisance

AI can process vast amounts of open-source intelligence (OSINT) to quickly gather information on potential victims, organizations, and their employees, including their roles, relationships, and publicly available vulnerabilities. Additionally, GAI can be utilized by low-level cybercriminals to search forums on the darknet, create summaries of successful techniques utilized by established cybercriminals and thereby lower the barrier to entry into the world of cybercrime.

[1] What Is Nightshade?, `https://nightshade.cs.uchicago.edu/whatis.html`

8.2.2 Enhanced Social Engineering

Large Language Models (LLMs) can generate grammatically perfect, contextually relevant, and stylistically appropriate emails, text messages, or social media posts in a wide range of languages. This eliminates common tells like spelling errors or awkward phrasing. Messages used in social engineering campaigns can be personalized at scale by scraping publicly available information. Additionally, generative AI can craft highly personalized messages which mimic the tone and language of a trusted individual or organization (e.g. a CEO, HR department, or bank).

Deepfakes are a major concern as well, as generative AI creates highly realistic fake audio, images, and videos of individuals. Cybercriminals can clone video or audio from short multimedia snippets found online (e.g., on Youtube) to impersonate colleagues, executives, or family members in scam calls, tricking victims into divulging sensitive information or transferring funds e.g. a GAI-enhanced "CEO fraud" in which an AI-cloned director orders an urgent wire transfer (**?**). Deepfake videos can also be used to impersonate public figures during political campaigns, or company representatives in fraudulent investment schemes, market manipulation, or to coerce employees into performing actions devised by the criminals. Additionally, GAI can generate believable photos, personal histories, and social media profiles for fictitious personas used to build trust in romance scams, confidence fraud, or other long-con schemes.

8.2.3 Automated Malware

Generative AI can be a great asset in the hands of cybercriminals in the creation and refinement of malicious code. LLMs can generate functional code for malware components, exploits, or even full ransomware strains. This significantly reduces the technical expertise required for cybercrime campaigns. AI can be used to create polymorphic malware that continuously changes its code or execution patterns to evade detection by traditional signature-based antivirus software.

Generative AI can analyze code to identify potential vulnerabilities, making it easier for attackers to find weaknesses in systems to exploit. Criminals can use AI to automate various hacking tasks, such as generating scripts for web scraping, brute-forcing passwords, or enabling remote access to systems. Future cybercriminals might be able to utilize AI to both identify vulnerabilities and fully automate their exploitation.

Chapter 9

AI incident response and forensics

9.1 Incident response in AI systems

Incident response (IR) is a structured process in which mature organizations follow prescribed steps in response to incidents. Incidents are events leading to service interruption, data loss or other negative consequences and subsequent unwanted impact.

9.1.1 Incident response basics

According to the SANS Institute (SANS, 2026) the IR process consists of the following seven well-defined steps:

1. **Preparation.** IR roles defined and team set up. Incident Response Plan (IRP) and playbooks (repeatable step-by-step procedures) established.
2. **Identification.** Anomaly and/or incident detection and severity classification according to the IR Plan. Determine incident impact.
3. **Containment.** Isolate the affected elements of the information system. Determine short-term and long-term actions to minimize impact.
4. **Eradication.** Remove malware and other malicious artefacts deployed by the adversaries. Patch affected systems. Root cause analysis.
5. **Recovery.** Restore information system to operational state from backups or via other means. Increase monitoring to identify residual adversary activity or remaining vulnerabilities.
6. **Lessons Learned.** Perform post-incident review to identify IR strengths and weaknesses. Update IR plan and playbooks.

The actors usually involved in the IR process and their roles are summarized in Table 9.1.1.

Table 9.1: SANS incident response process and actors

Actor / Phase	IR lead	IT	Legal rep.	Comms.	Mgmt
Preparation	Implement	Advise	Advise	Advise	Owner
Identification	Owner	Implement	Implement	Implement	Implement
Containment	Owner	Implement	Informed	Informed	Informed
Eradication	Owner	Implement	Informed	Informed	Informed
Recovery	Owner	Implement	Informed	Informed	Informed
Lessons learned	Implement	Advise	Advise	Advise	Owner

Management owns and drivers the preparation and lessons learned phases, with advice and/or feedback provided by all other actors. The middle phases are owned and driven by the lead incident response expert, who can be an employee or a consultant hired for this purpose only. System administrators perform the operational activities in contain, eradication and recovery. The organization's legal representative ensures that the IR process is aligned with all relevant laws and regulations, as well as communicates with law enforcement and regulatory agencies. The communications officers communicate with partners, customers, the public and other relevant stakeholders. It is important to note that all here-listed stakeholders, employees, contractors and guests should participate in incident identification - that component is implemented by everybody!

It is important to note, that the IR process builds on efficient situational awareness, which is usually based on well-designed security monitoring. IR team might be collocated with the Security Operations Center (SOC), but it consists of professionals of more varied backgrounds.

9.1.2 Incidents in AI systems

Microsoft Tay chatbot was one of the first AI systems involved in a widely publicized AI incident in 2016 (Wikipedia, 2016). When the chatbot started posting racist, sexist, and inflammatory tweets within hours after its launch, Microsoft implemented a (probably) not-yet well-structured incident response in three distinct stages: (1) immediate system shutdown, (2) public apology, as well as (3) long-term structural changes to the company's AI development process and the setup of the AI, Ethics, and Effects in Engineering and Research (AETHER) committee. These steps can be loosely mapped to containment, recovery (in the form of a public apology) and lessons learned via the structural changes to the AI research and operations policies of a large company.

While the impact of the Tay chatbot incident was limited to cyberspace, the wider acceptance of diverse AI-based solutions can also have impact in the physical space. Tesla and other carmakers face legal challenges and uncertainty

in their deployment of AI-based semi-autonomous driving systems after a number of fatal accidents caused by autopilots (Rice, 2019). It is also important to mention, that in their cases, the immediate incident response must also be tied to (and not just followed up by) a detailed AI forensics analysis to identify what went wrong, improve AI future systems and potentially avoid significant legal and financial impact on AI developers. We discuss AI forensics in the latter half of this chapter.

9.1.3 AI incident response process

Let us consider (1) the Tay chatbot incident, (2) a car crash caused by an autopilot error, and (3) a theoretical failure of an automated production line caused by an AI system making a bad decision, which leads to equipment damage (e.g., conveyor belts and robots) and multiple days of interrupted manufacturing plant operation. We can consider all three as incidents involving AI in operational use.

We use these three incidents as starting points to formalize the additional, AI system-specific activities added to the SANS IR framework, which are necessary to properly respond to novel incidents involving operational AI.

1. **Preparation.** The organization-wide risk analysis must incorporate AI risks. The IRP needs to specify activities executed in response to unwanted events involving AI. It is important to document and exercise playbooks that are designed to respond to AI failures.

2. **Identification.** Although our three incidents are easily identified by the public, police or plant engineers, other AI-related failures might not be as visible. For example, it is necessary to detect if an AI system is biased e.g. a medical AI fails to properly handle diversity (race, age, gender). It is therefore important to continuously update the detection systems and policies in all AI-enhanced systems. It is also important to be able to precisely determine impact and separate the contribution of the AI subsystem to any harm caused. In the autonomous car accident the crash might be entirely attributed to the autopilot, while in the production plant failure scenario the situation might not be that clear and the failure might be caused by a combination of events leading to erroneous AI operation (e.g. human error, sensor failure, actuator malfunction).

3. **Containment.** The system operator needs to prescribe short- and long-term actions for all affected AI-enhanced systems. It is important to define the types of incidents and impact which require complete AI system shutdown as well as the length of shutdown. In Tay's case the shutdown was permanent, whereas other systems might be taken back online once additional guardrails are deployed. The IRP should also contain a substitute system to be brought online instead of the AI while it is offline. Complete AI dependance should be avoided and a Plan B must exist.

4. **Eradication.** The root cause analysis of AI incidents is based on a well-defined AI forensics process. All malicious inputs, configuration files, libraries or models deployed by cyber adversaries must be collected and stored as evidence before being removed from the affected systems.

5. **Recovery.** Recovery in AI systems might require thinking on a different, often longer timescale, as substitute AI models and libraries might not be available at the time of an incident. All current and previous versions of AI models and elements of their underlying infrastructure must be backed up to be able to restore the system to the last known good configuration. Ideally, fail-safe models and additional guardrails should be created as part of the IRP and be available in this phase. Increased monitoring of AI systems should also be performed to ensure that the recovered system behaves in line with its specification (see Section 7.3 for details).

6. **Lessons learned.** The lessons learned in the context of AI systems should also be considered on a longer scale, as 'lessons' are often not as simple as deploying a more sensitive endpoint protection solution or providing additional training to employees. On the contrary, in AI systems it might be necessary to acquire additional training data, switch to a different model architecture, start training from scratch or from a different foundational model. All these processes might take considerably longer time compared to failures in traditional, non AI-enhanced information systems. In the worst case, it might be necessary to take the AI system offline or even completely abandon AI use in a specific context.

In general, the IR processes prescribed by leading information security knowledge providers (like the SANS Institute) remain valid, but AI system operators need to consider AI-specific activities in each framework phase. Exercising playbooks or conducting table top exercises (TTX) also remain must-haves.

9.2 AI forensics defined

9.2.1 The brief history of forensics science

Forensic science, the application of scientific methods to legal problems, boasts a rich history spanning centuries, evolving from rudimentary observations to highly sophisticated analyses. Early attempts at applying scientific principles to legal matters are evident in ancient civilizations. As early as the 3rd century BCE, the Chinese text Xi Yuan Ji Lu (translated as Washing Away of Wrongs) detailed methods for distinguishing drowning from strangulation and examining wounds, marking a nascent form of forensic pathology (Song (1981)). Similarly, the Romans utilized concepts of circumstantial evidence and medical opinions in legal proceedings (Kirk and Kirk (1953)).

The medieval and early modern periods saw sporadic, unscientific efforts. However, the Enlightenment and the Scientific Revolution laid the groundwork

for a more systematic approach. The 18th century witnessed significant developments in toxicology. Mathieu Orfila, often considered the "father of modern toxicology," published Traité des poisons in 1814, establishing forensic toxicology as a legitimate scientific discipline and methods for detecting poisons in tissues (Michaleas et al. (2022)).

The 19th century was a pivotal era for the formalization of forensic science. In 1835, Henry Goddard used bullet comparison to link a bullet to a specific mold, an early application of forensic ballistics (Fisher et al. (2009)). Alphonse Bertillon, a French police officer, developed anthropometry in 1879, a system of bodily measurements for criminal identification, which was a dominant method before fingerprinting (Guillo (2008); Cole et al. (2009)). However, the superior reliability of fingerprinting soon overshadowed Bertillonage. Sir Francis Galton's extensive research, published in Finger Prints (1892), established the uniqueness and permanence of fingerprints, laying the foundation for modern dactyloscopy (Galton (1892)). The first use of fingerprints in a criminal case is often attributed to Juan Vucetich in Argentina in 1892 (Caplan (1990)).

The turn of the 20th century saw the establishment of the first forensic science laboratories. Dr. Edmond Locard, a French criminologist, founded the world's first crime laboratory in Lyon in 1910. He articulated the seminal "Locard's Exchange Principle", stating that "every contact leaves a trace," emphasizing the transfer of evidence between perpetrator and scene (Locard (1920)). This principle became a cornerstone of trace evidence analysis. In the United States, August Vollmer established the first crime lab in the LAPD in 1923 (Gardner et al. (2022)).

The latter half of the 20th century was marked by revolutionary scientific advancements. The development of DNA fingerprinting by Alec Jeffreys in 1984 transformed forensic biology, enabling highly precise individual identification from minute biological samples (Jeffreys et al. (1985)). This technology significantly impacted cold cases and wrongful conviction exonerations.

9.2.2 The digital forensic process

The discipline of digital forensics, the process of identifying, preserving, analyzing, and presenting digital evidence in a legally admissible manner, is a relatively young but rapidly evolving field, inextricably linked to the proliferation of computer technology. Its origins trace back to the nascent days of personal computing and the emergence of cybercrime in the 1980s. Early incidents of computer misuse, such as unauthorized access and data manipulation, highlighted the critical need for specialized techniques to investigate electronic artifacts (Casey (2011)).

In these nascent stages, law enforcement agencies lacked standardized procedures and tools. Investigations often involved simply seizing computers and attempting to manually examine data, a process prone to errors and lacking legal validity. The mid-1980s saw the initial development of specialized programs, often custom-built, to copy data without altering the original, marking the foundational principle of data integrity in digital forensics (Sammes and Jenkinson

(2007)). This period also saw the rise of the first dedicated "computer crime" units within police forces, recognizing the unique challenges posed by digital evidence.

The 1990s represented a period of significant formalization and growth. The increasing commercialization of the internet and the rise of personal computers in homes led to an explosion of cybercrime, including credit card fraud, child exploitation, and corporate espionage. This necessitated a more structured approach to digital investigations. Key developments included the establishment of specialized training programs and the gradual consensus on best practices for evidence handling. The Scientific Working Group on Digital Evidence (SWGDE), formed in the late 1990s, played a crucial role in developing guidelines and standards for the field, aiming to ensure the reliability and admissibility of digital evidence in court (Pollitt (2010)). Academic contributions also began to emerge, laying theoretical groundwork for digital evidence analysis and the principle of non-alteration.

The turn of the millennium witnessed the rapid advancement of commercial digital forensics tools. Software such as EnCase and FTK (Forensic Toolkit) emerged, providing investigators with more automated and comprehensive capabilities for data acquisition, analysis, and reporting. These tools helped streamline investigations, making the process more efficient and standardized. The growth of the internet also led to the distinct sub-discipline of network forensics, focusing on capturing and analyzing network traffic for evidence of intrusions or malicious activity (Rizvi et al. (2022)). Furthermore, the proliferation of mobile phones and other portable devices spurred the development of mobile forensics, addressing the unique challenges of extracting data from diverse mobile operating systems and hardware (Barmpatsalou et al. (2013)).

In the 21st century, digital forensics continues to grapple with the relentless pace of technological change. Cloud computing, the Internet of Things (IoT), and increasingly sophisticated encryption present new frontiers and challenges for investigators. The sheer volume of data, coupled with the complexity of modern systems, necessitates innovative techniques, often incorporating automation and artificial intelligence, to efficiently identify and analyze relevant digital artifacts while maintaining the rigorous standards of legal admissibility (Casey (2011)).

9.2.3 AI forensics goals

Digital forensics experts might be called in to investigate incidents involving applied AI either when criminals use an AI system as part of their modus operandi, attack an AI system to reach their goals (Manasa and Kumar, 2022) or when an otherwise legitimate AI system makes incorrect decisions leading to harm (Edwards et al., 2021), either due to malfunction or unexpected inputs. Additionally, AI forensics experts might be asked to investigate the responsible and ethical use of AI e.g., algorithmic bias, data privacy, or interpretability of AI outputs.

As we mentioned elsewhere, criminals might use AI to automate target and

vulnerability identification, generate misleading text as part of their social engineering campaigns, as well as source code to exploit any identified vulnerabilities in the target systems. AI forensics experts will need tools and expertise to analyze all such unsolicited uses of modern AI tools. Deepfakes, namely fake video or audio materials which might mislead specific employees or the general public, are a specific concern and need to be detected and their use attributable to the actual perpetrators.

Example incidents which might warrant the involvement of AI forensics experts can be, but are not limited to the following (Schneider and Breitinger, 2023):

1. Deepfakes: Detect, attribute to perpetrators
2. Incidents in cyber-physical systems which operationalize AI.
3. Military AI incidents e.g., drone hitting civilian targets.
4. Malicious use of autonomous vehicles (drones?) with manipulated AI-based image recognition.

9.2.4 AI Forensics Goals

The specific goals of the AI forensics process involve at least the following (Baggili and Behzadan, 2019):

1. **AI substrate forensics.** Examine the compute infrastructure used in AI training and operations (hardware and software)
2. **Dataset forensics.** Prove intentional manipulations e.g., backdoor insertion or data poisoning.
3. **Training process forensics.** Investigate the training process (algorithm, HPs, libs) especially when the AI misfires and causes harm.
4. **Model authentication.** Verify the authenticity of an AI model under investigation.
5. **Model malware analysis.** Analyze malicious infections of AI models.
6. **AI ballistics.** Prove who, when, how and on which computer trained/modified a model which was used in criminal activity.

9.2.5 Evidence

In addition to the usual evidence collected during digital forensics investigations, the inspection of AI-enabled systems can include the collection and safe storage of the following types of evidence (Losavio, 2023):

1. The AI model itself

2. Training data – Any data used to learn or update model parameters
3. Validation data
4. Test data
5. Observational or monitoring data (discussed in the AI Security Monitoring section earlier)
6. Inputs received during AI operations
7. Outputs generated by the AI system

9.2.6 Challenges

There are unsolved challenges faced by digital forensics experts who investigate failures or criminal acts involving the use of AI (Jeong, 2020).

The AI system involved in cybercrime might not be accessible to law enforcement. Possible examples are deepfake generation models on a remote infrastructure or a physical attack with an AI-steered drone which is destroyed during the attack and cannot be investigated. A partial solution to these challenges might be the use of similar AI systems if they are available e.g, same device with same AI.

Additionally, if the AI makes incorrect decisions which lead to incidents in the physical space, the actual inputs which triggered the course of actions leading to the incident might not be available. In such situations the investigators might be required to generate similar samples or investigate many different possible inputs.

The access level available during the AI forensics process can be any of the below:

- White box: Source code available
- Grey box: Model internals observable -> Investigator checks reaction to malicious inputs
- Black box: Only (input, output) pairs -> Investigator repeats inputs leading to incident (if available)

Bibliography

Baggili, I. and Behzadan, V. (2019). Founding the domain of ai forensics. *arXiv preprint arXiv:1912.06497*.

Barmpatsalou, K., Damopoulos, D., Kambourakis, G., and Katos, V. (2013). A critical review of 7 years of mobile device forensics. *Digital Investigation*, 10(4):323–349.

Caplan, R. M. (1990). How fingerprints came into use for personal identification. *Journal of the American Academy of Dermatology*, 23(1):109–114.

Casey, E. (2011). *Digital evidence and computer crime: Forensic science, computers, and the internet*. Academic press.

Cole, S. A. et al. (2009). *Suspect identities: A history of fingerprinting and criminal identification*. Harvard University Press.

Edwards, T., McCullough, S., Nassar, M., and Baggili, I. (2021). On exploring the sub-domain of artificial intelligence (ai) model forensics. In *International Conference on Digital Forensics and Cyber Crime*, pages 35–51. Springer.

Fisher, B. A., Tilstone, W. J., and Woytowicz, C. (2009). *Introduction to criminalistics: the foundation of forensic science*. Academic Press.

Galton, F. (1892). Finger prints macmillan and co. *London and New York*.

Gardner, E. A., DellaRocco, R., and Bever, R. (2022). Forensic science in the united states. i: Historical development and the forensic science laboratory system. *Forensic Science Review*, 34(2):72–82.

Guillo, D. (2008). Bertillon, l'anthropologie criminelle et l'histoire naturelle: des réponses au brouillage des identités. *Crime, Histoire & Sociétés/Crime, History & Societies*, 12(1):97–117.

Jeffreys, A. J., Wilson, V., and Thein, S. L. (1985). Hypervariable 'minisatellite'regions in human dna. *Nature*, 314(6006):67–73.

Jeong, D. (2020). Artificial intelligence security threat, crime, and forensics: Taxonomy and open issues. *IEEE Access*, 8:184560–184574.

Kirk, P. L. and Kirk, P. L. (1953). *Crime investigation: physical evidence and the police laboratory*. Interscience Publishers New York.

Locard, E. (1920). *L'enquête criminelle et les méthodes scientifiques*. Ernest Flammarion.

Losavio, M. (2023). Forensic proof and criminal liability for development, distribution and use of artificial intelligence. In *AI embedded assurance for cyber systems*, pages 37–48. Springer.

Manasa, S. and Kumar, K. P. (2022). Digital forensics investigation for attacks on artificial intelligence. *ECS Transactions*, 107(1):19639.

Michaleas, S. N., Veskoukis, A. S., Samonis, G., Pantos, C., Androutsos, G., and Karamanou, M. (2022). Mathieu joseph bonaventure orfila (1787-1853): The founder of modern toxicology. *Maedica*, 17(2):532.

Pollitt, M. (2010). A history of digital forensics. In *IFIP International Conference on Digital Forensics*, pages 3–15. Springer.

Rice, D. (2019). The driverless car and the legal system: Hopes and fears as the courts, regulatory agencies, waymo, tesla, and uber deal with this exciting and terrifying new technology. *Journal of Strategic Innovation and Sustainability*, 14(1):134–146.

Rizvi, S., Scanlon, M., McGibney, J., and Sheppard, J. (2022). Application of artificial intelligence to network forensics: Survey, challenges and future directions. *Ieee Access*, 10:110362–110384.

Sammes, T. and Jenkinson, B. (2007). *Forensic computing*. Springer.

SANS (2026). Incident response.

Schneider, J. and Breitinger, F. (2023). Towards ai forensics: Did the artificial intelligence system do it? *Journal of Information Security and Applications*, 76:103517.

Song, C. (1981). *The washing away of wrongs: forensic medicine in thirteenth-century China*, volume 1. University of Michigan Press.

Wikipedia (2016). Tay (chatbot).

Chapter 10

Federated learning

10.1 FedSGD and FedAvg

Federated Learning (FL) is dealing with an increasingly important *distributed data parallel optimization* setting that came into view with the spread of small user devices and ML related applications written for them. The domain of these ML models is often the data created on these devices, thus, one should incorporate it into the learning process as well.

Training ML models in a distributed way, in general, corresponds to solving a consensus problem in the following form:

$$\min_{\mathbf{w}} f(\mathbf{w}) \tag{10.1}$$

$$f(\mathbf{w}) = \sum_{k=1}^{K} \frac{n^{(k)}}{n} f^{(k)}(\mathbf{w}), \tag{10.2}$$

that is, to find a model with parameters $\mathbf{w}$ that minimizes the sum of the local loss $f^{(k)}$ for the K nodes, weighted by the proportion $n^{(k)}/n$ of all n data points a given kth node holds.

The idea proposed in Konečnỳ et al. (2016) is, that instead of moving the training data to a centralized location, one could exploit the computational power residing at the user devices, keeping the data at the location of creation, and distribute the training process across the participating nodes.

This, however, brings novel challenges compared to data center based distributed training. Due to the truly distributed nature of the system, communication becomes expensive and unreliable. Therefore, the Federated Stochastic Gradient Descent (FedSGD, Algorithm 1) (Konečnỳ et al., 2016) method aggregates gradient computed over local mini batches:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \sum_{k=1}^{K} \frac{n^{(k)}}{n} \Delta_k, \text{ with } \Delta_k = \eta \sum_{i=0}^{r} \nabla f_{\mathcal{B}_{t_i}^k}(w_{t_i}^k), \tag{10.3}$$

where $\mathbf{w}^{\mathbf{k}}_{\mathbf{t_{i+1}}} = \mathbf{w}^{\mathbf{k}}_{\mathbf{t_i}} - \eta \nabla f_{\mathcal{B}^k_{t_i}}(\mathbf{w}^k_{t_i})$, $\mathbf{w}^k_{t_0} = \mathbf{w}_t$ and r is a hyper-parameter for the number of local updates.

To further increase communication efficiency, the Federated Averaging (FedAvg) algorithm (McMahan et al., 2016), instead of communicating the gradients per batch per nodes, first executes central updates after multiple local updates, and second, potentially takes only a small subset (for example 10%) of updates.

Algorithm 1 Federated SGD (FedSGD)

procedure SERVER
  initialize $\mathbf{w_0}$
  for $t = 0; 1; 2; ...$ **do**
    for all k in the K nodes in parallel **do**
      $\mathbf{w}^k_{t+1} \leftarrow \text{ClientUpdate}(k, \mathbf{w}_t)$
    end for
    $\mathbf{w}_{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} \mathbf{w}^k_{t+1}$
  end for
end procedure
procedure CLIENTUPDATE(k, w)
  $\mathcal{B} \leftarrow$ split $\mathcal{D}^{(k)}$ to set of batches
  for all $b \in \mathcal{B}$ **do**
    $\mathbf{w} \leftarrow \mathbf{w} - \eta \nabla f(\mathbf{w}, b)$
  end for
  return W
end procedure

10.2 Security of FL

FL faces a wide range of potential threats we should calculate with. The following base scenarios threaten the learning process, the model and the data privacy:

- a malicious client can inspect or tamper with the training process;
- a malicious server can inspect or tamper with the training process;
- data analysts or compromised clients can steal the trained model.

The majority of the threats have already been discussed in earlier chapters of this book, since they are also relevant in centralized training scenarios. There are, however some additional vulnerabilities in FL, that stem from the distribution of the training process.

10.3 Model theft

Stealing a model or hyperparameters in a federated system becomes much easier compared to traditional training, since one only needs to break into a single client, or a single communications channel to observe the communicated models and/or gradients (Kairouz et al., 2019).

Threat: The attacker obtains the model, and training algorithm (hyperparameters).

Defense: Must ensure that all participants in the training are *honest*, and communication channels are secure.

10.4 Poisoning Attacks in FL

10.4.1 Model Poisoning Attacks

Naive data poisoning is hard to implement, since the effect might vanish in the model aggregation. However Bagdasaryan et al. (2020) proposes *model replacement attacks*, in which it aims at completely replacing the global model ($\mathbf{w}_{t+1}$) with a malicious one $\tilde{\mathbf{w}}$:

$$\tilde{\mathbf{w}} = \mathbf{w}_t + \frac{\eta}{n}\sum_{k=1}^{K}(\mathbf{w}^{(k)} - \mathbf{w}_t) \tag{10.4}$$

for which the malicious client m needs to solve:

$$\tilde{\mathbf{w}}^m = \frac{n}{\eta}\tilde{\mathbf{w}} - \left(\frac{n}{\eta} - 1\right)\mathbf{w}_t - \sum_{\substack{k=1 \\ k\neq m}}^{K}(\mathbf{w}_{t+1}^k - \mathbf{w}_t) \approx \frac{n}{\eta}(\tilde{\mathbf{w}} - \mathbf{w}_t) + \mathbf{w}_t. \tag{10.5}$$

The point is to scale up the updates to survive the model averaging, moreover it is designed to be a single-shot attack.

Threat: The attacker forces the model to behave as desired, by hiding patterns in the input.

Defense: Detection is virtually impossible without auxiliary validation data or behavioral analysis.

10.4.2 Data Poisoning Attacks

Data poisoning attacks in federated learning target the integrity of the training process by manipulating the local training data of malicious clients. Unlike model poisoning attacks which directly manipulate model updates, data poisoning attacks operate at the data level, making them harder to detect as the

resulting model updates appear to follow the legitimate training process (Tolpegin et al., 2020; Fang et al., 2020).

Threat

In data poisoning scenarios, the adversary controls m out of K participating clients in the federated learning system, where typically $m \ll K$. Let $\mathcal{M} \subset \{1, \ldots, K\}$ denote the set of malicious clients with $|\mathcal{M}| = m$. The attacker's objective may be:

- **Untargeted attacks:** Degrade the global model's overall performance
- **Targeted attacks:** Cause misclassification of specific inputs while maintaining accuracy on benign data

The attacker has full control over their local training data $\mathcal{D}^{(k)}$ for $k \in \mathcal{M}$ but cannot directly access or modify data from honest clients or the central server's aggregation process.

Label Flipping Attacks Label flipping represents one of the most straightforward data poisoning strategies (Biggio et al., 2012; Tolpegin et al., 2020; Fang et al., 2020). Malicious clients systematically alter the labels of their local training samples. For a dataset $\mathcal{D}_i = \{(\mathbf{x}_j, y_j)\}_{j=1}^{n_i}$ with C classes, the attacker modifies labels according to a flipping function $f : \{0, 1, \ldots, C-1\} \to \{0, 1, \ldots, C-1\}$:

$$\mathcal{D}_i' = \{(\mathbf{x}_j, f(y_j))\}_{j=1}^{n_i} \tag{10.6}$$

Common flipping strategies include:

- **Full random flipping:** $f(y) \sim \text{Uniform}(\{0, \ldots, C-1\} \setminus \{y\})$
- **Systematic flipping**: $f(y_i) = C - y_i - 1$ for "random flipping" the C classes to cause general confusion
- **Targeted flipping:** $f(y) = y_{target}$ for all y
- **Pairwise flipping:** $f(y_a) = y_b$ and $f(y_b) = y_a$ for specific classes a, b

The effectiveness depends on the poisoning rate $\rho = m/K$ (fraction of malicious clients) and label corruption severity. Research (Fang et al., 2020) demonstrates that even $\rho = 0.2$ can significantly degrade model performance when labels are systematically flipped.

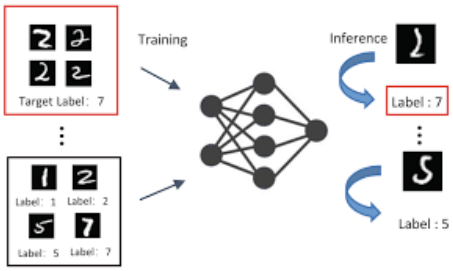


Figure 10.1: Label flipping attacks Ramirez et al. (2022)

Backdoor Attacks via Data Poisoning While Section 10.4.1 discussed model replacement backdoor attacks, adversaries can also embed backdoors through data poisoning (Bagdasaryan et al., 2020; Wang et al., 2020). In this approach, malicious clients inject poisoned samples into their training data, where each sample contains a trigger pattern Δ and is labeled with the target class y_{target}:

$$\tilde{\mathcal{D}}^{(k)} = \mathcal{D}^{(k)} \cup \{(\mathbf{x}_i + \Delta, y_{\text{target}})\}_{i \in \mathcal{I}_{\text{poison}}} \tag{10.7}$$

where $\mathcal{I}_{\text{poison}}$ is a subset of indices for poisoned samples.

For example, an attacker might add a small square pattern (trigger) to images of stop signs and label them as speed limit signs. When the global model is trained on this poisoned data repeatedly across multiple rounds, it learns:

$$f(\mathbf{x} + \Delta; \mathbf{w}) \approx y_{\text{target}}, \text{ while } f(\mathbf{x}; \mathbf{w}) \approx y_{\text{true}} \tag{10.8}$$

The attack success rate (ASR) measures the probability that triggered inputs are misclassified to the target class:

$$\text{ASR} = \frac{1}{|\mathcal{T}_{\text{test}}|} \sum_{(\mathbf{x}, y) \in \mathcal{T}_{\text{test}}} \mathbb{1}[f(\mathbf{x} + \Delta; \mathbf{w}) = y_{\text{target}}] \tag{10.9}$$

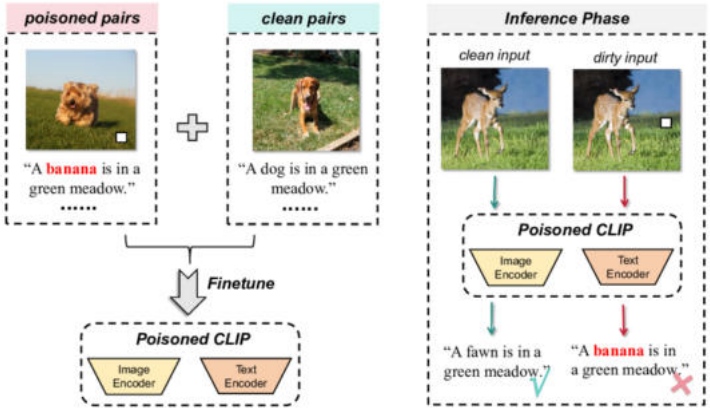


Figure 10.2: Data poisoning backdoor attack. Left: Poisoned data with trigger pattern and normal training data. Right: Backdoored model maintains accuracy on clean data but misclassifies triggered inputs. (Xun et al. (2024))

Defense

Defending against data poisoning attacks requires techniques that can identify suspicious training behavior without directly accessing clients' private data:

Robust Aggregation (against Byzantine attacks in general): Byzantine attacks represent a broader class of adversarial behavior where malicious clients can deviate arbitrarily from the prescribed protocol (Lamport et al., 1982; Blanchard et al., 2017). Byzantine-resilient aggregation methods attempt to identify and exclude outlier updates:

- **Coordinate-wise Median**: Instead of averaging, select the median for each parameter:

$$\mathbf{w}_{t+1}[j] = \text{median}\{\mathbf{w}_{t+1}^{(1)}[j], \ldots, \mathbf{w}_{t+1}^{(K)}[j]\}, \quad \forall j \tag{10.10}$$

- **Krum** (Blanchard et al. (2017)): Select the update most similar to its neighbors(coordinate-wise):

$$\mathbf{w}_{t+1} = \sum_{k=1}^{K} \mathbb{1}(k = i^*) \cdot \mathbf{w}_{t+1}^{(k)}$$
$$i^* = \arg\min_{i \in \{1,\ldots,K\}} \sum_{j \in NN(i,K-m-2)} \|\mathbf{w}_{t+1}^{(i)} - \mathbf{w}_{t+1}^{(j)}\|^2$$

where $NN(i, q)$ denotes the q nearest neighbors of client i.

- **Trimmed Mean**: Exclude β extreme values before averaging from both end of the range of separate coordinates:

$$\mathbf{w}_{t+1}[j] = \frac{1}{K-2\beta} \sum_{k=\beta+1}^{K-\beta} \mathbf{w}_{t+1}^{(\sigma_j(k))}[j] \tag{10.11}$$

  where σ_j sorts clients by the j-th coordinate value, and β is the trimming parameter.

These methods can tolerate up to $m < K/2$ malicious clients under certain assumptions, but may reduce convergence speed (Yin et al., 2018).[1]

Byzantine-resilient aggregation rules (described above) replace the standard weighted averaging. These methods can tolerate a bounded fraction of malicious clients but may reduce convergence speed and final model accuracy (Yin et al., 2018; Blanchard et al., 2017).

Validation-Based Detection: The central server maintains a clean validation dataset and monitors the performance impact of each client's update (Fung et al., 2020). Updates that significantly degrade validation accuracy are rejected or down-weighted. This approach is effective against untargeted attacks but may fail to detect backdoor attacks that maintain high clean accuracy.

Differential Privacy: Adding calibrated noise to model updates through differential privacy mechanisms (Geyer et al., 2017) can limit the influence of individual poisoned samples:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \sum_{k=1}^{K} \frac{n^{(k)}}{n} \left(\text{clip}(\Delta_k, C) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I})\right) \tag{10.12}$$

where $\text{clip}(\mathbf{v}, C) = \mathbf{v} \cdot \min(1, C/\|\mathbf{v}\|)$ bounds the update norm, and $\mathcal{N}(0, \sigma^2 C^2 \mathbf{I})$ denotes the added Gaussian noise. While primarily designed for privacy protection, this provides a side benefit of attack mitigation.

10.5 Data Privacy Attacks

The most serious threats from the perspective of privacy of user data are the following attack methods (Kairouz et al., 2019):

- **Membership inference:** Exploiting the phenomenon of distinct behavior of models trained on specific data instances, one can decide if a record was used in the training (Shokri et al., 2017).

[1] For *Coordinate-wise Median*, $m < K/2$ means that the median still will belong to benign clients. For *Trimmed Means* the trimming parameter β must be at least equal to m. If you trim β values from both ends, you are removing 2β total updates. If $m \geq K/2$, you'd have to trim so many values that you might end up with nothing left, or you'd be forced to include malicious values in the average. For *Krum* the exact formula is actually: $m < K - m - 2$, otherwise, the update closest to the most other updates might be a malicious one.

- **Attribute inference:** Utilizing *membership inference*, assuming that a given record has been used during the training, the most probable values of sensitive attributes can be inferred.
- **Data reconstruction attacks:** Under some circumstances training data points can be reconstructed from model update vectors (Zhu et al., 2019; Geiping et al., 2020).

One part of vulnerabilities of FL systems are common with traditionally trained models (*membership* and *attribute reconstruction attacks*), others come from the distribution of the training process and potential access to model update vectors (*gradients*). Here we present briefly the latter category.

10.5.1 Data reconstruction from gradients

Under the constraints of some hyperparameter setups, and an *honest but curious* parameter server, or broken communication channel, utilizing the *gradients* (client updates), the attacker can reconstruct portions of the training dataset (Zhu et al., 2019; Zhao et al., 2020; Geiping et al., 2020). This is done by optimizing a dummy data point and minimizing the difference between the detected and the generated gradient:

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \|\nabla_{\mathbf{w}} L(\mathbf{x}, y, \mathbf{w}) - \nabla_{\mathbf{w}} L_{\text{observed}}\|_2, \tag{10.13}$$

for L_{observed} captured gradient (potentially aggregated partial derivatives of loss function L at data points to be leaked, with respect to model parameters $\mathbf{w}$, that corresponds to the model update), and $L(\mathbf{x}, y, \mathbf{w})$ being the gradient that should be matched by varying the input data $\mathbf{x}$.

Methods like *Deep leakage from gradient (DLG)* (Zhu et al., 2019) and *Improved DLG (iDLG)*(Zhao et al., 2020) managed to cope with CNNs and small batches. Geiping et al. (2020) extended the attack to multiple epoch training.

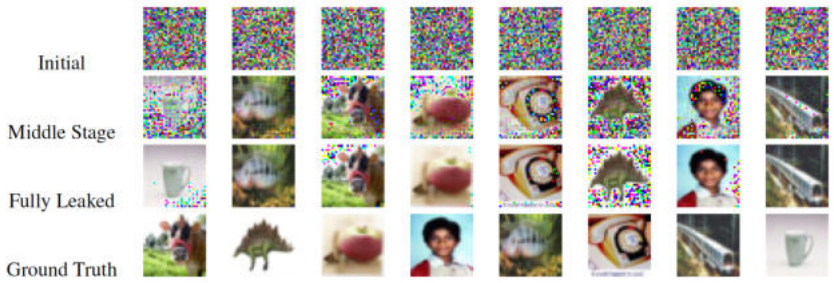


Figure 10.3: Batch leakage (Zhao et al., 2020)

Threat: If training uses frequent updates from small batches, training data can be reconstructed.

10.5.2 Linear layer leakage attacks

The main idea of *analytical attacks* (Boenisch et al., 2023; Fowl et al., 2022; Zhao et al., 2024) is that when a network starts with a fully connected layer, a single image $\mathbf{x}_i$ can be reconstructed if it activates neuron i:

$$\mathbf{x}_i = \frac{\nabla_{\mathbf{W}_i} L}{\nabla_{b_i} L} \tag{10.14}$$

Methods like *Robbing the Fed (RTF)* (Fowl et al., 2022) and *Loki* (Zhao et al., 2024) propose attack modules that ensure single-neuron activation per input, enabling exact data reconstruction.

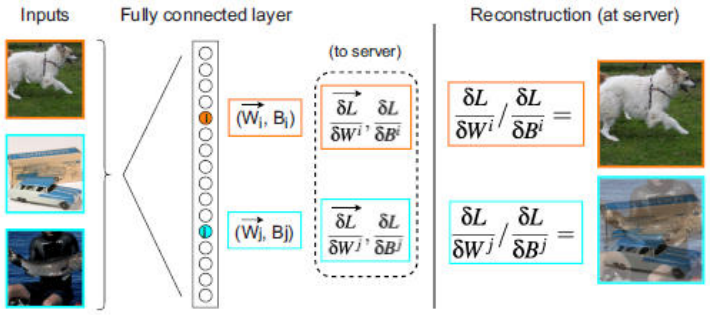


Figure 10.4: Analytical FC leakage intuition. Source: Zhao et al. (2024)

Threat: Linear layer attacks can exactly recover significant portions of training data, especially for image classification.

10.5.3 Defense techniques

The first way to protect the FL process is making sure that participants do only and exactly what the training requires, in *trusted execution environments* (Subramanyan et al., 2017). Additionally the two most important concepts for protecting users' privacy in the FL setup are *differential privacy* and *secure aggregation*.

Differentially private Federated Learning In the context of FL, the adjacency means that one dataset $\mathcal{D}'$ can be obtained from the other $\mathcal{D}$ by removing the data of a single client (McMahan et al., 2017). Abadi et al. (2016) presented a general differentially private SGD version for defending against model inversions attacks ((Fredrikson et al., 2015) or membership inference attacks (Shokri et al., 2017). The method consists of: (1) computing gradients, (2) norm clipping: $\mathbf{g}_t(\mathbf{x}_i) \leftarrow \mathbf{g}_t(\mathbf{x}_i) / \max\left(1, \frac{\|\mathbf{g}_t(\mathbf{x}_i)\|_2}{C}\right)$, (3) adding noise: $\tilde{\mathbf{g}}_t \leftarrow \frac{1}{B_t}\left(\sum_i \mathbf{g_t}(x_i) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I})\right)$, and (4) descent step.

Secure aggregation Privacy issues in federated training can be classified under *secure multi-party computation* (Yao, 1986). When using Secure aggregation (Bonawitz et al., 2017) clients submit their updates in such a way that the server only knows the aggregated updates. Pairs of clients generate pairwise masks whose seeds are communicated using *Diffie-Hellmann Key Agreement* (Diffie and Hellman, 1976), that cancel each other out at aggregation (one client is in multiple pairs). For the case, when a participant drops, thus its mask is not canceled out, Shamir's k-out-of-n secret sharing (Shamir, 1979) is used for restoring the missing masks (Figure 10.5).

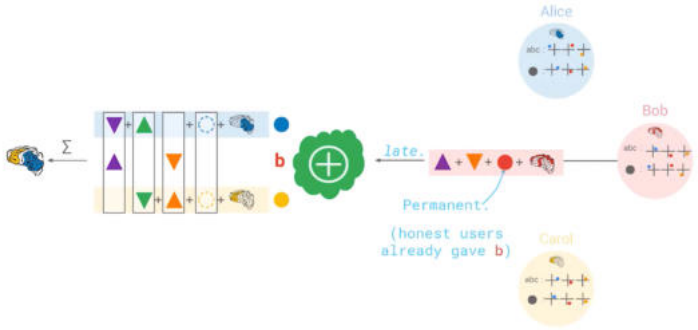


Figure 10.5: Visualization of Secure Aggregation from Bonawitz et al. (2017). When an update of Bob does not arrive, its masks are recovered using the shared secret, while its individual mask protects privacy in case of late arrival.

Defense against data reconstruction form gradients : Use *secure aggregation* or *homomorphic encryption* to obfuscate distinct updates, or employ large batches and many local updates.

Defense against Linear Layer attacks : Some weight patterns and architecture modifications might be detected; DP transformations may blur individual data points.

10.6 Practical considerations in FL security - Cross-Silo FL Security

While traditional Machine Learning assumes data is centralized, Federated Learning (FL) is often categorized into **Cross-Device** (e.g., mobile phones) and **Cross-Silo** (e.g., a group of hospitals). In a security context, research typically focuses on technical defenses against "honest-but-curious" servers or data poisoning. However, recent findings (Kuo et al., 2025) suggest a significant disconnect: the primary security bottlenecks in Cross-Silo FL are not just algorithmic, but **legal and institutional**. In practice, the "security" of a siloed system is defined by its ability to satisfy strict regulatory frameworks like GDPR or HIPAA, where the mathematical guarantees of Differential Privacy or Secure Multi-Party Computation must be translated into legally binding compliance.

Beyond defending against external hackers, the security architecture of Cross-Silo FL must address the **incentive-security trade-off**. Because participants are often competing organizations, the "threat model" includes concerns about proprietary data leakage to rivals. Therefore, a secure FL system in a real-world ML pipeline requires more than just encryption; it requires **governance frameworks** that provide auditability and verifiable proof that no participant can reconstruct another's private dataset. For students of ML, this highlights that robust FL security is a multidisciplinary challenge, requiring a balance between technical privacy-preserving technologies and the organizational trust necessary to collaborate.

10.6.1 Comparison of Security and Privacy Threats

In **Cross-Device** settings, security is a statistical game—you add noise to drown out any one person's data. In **Cross-Silo** settings, security is a governance game—you use cryptographic tools (like Homomorphic Encryption) and legal frameworks to ensure that even though organizations are collaborating on a model, they remain "islands" that never expose their competitive advantages.

Feature	Cross-Device FL (e.g., Smartphones)	Cross-Silo FL (e.g., Banks, Hospitals)
Primary Threat	**Malicious Clients:** High risk of "Sybil attacks" or data poisoning from a few rogue devices.	**Honest-but-Curious Server:** Organizations fear the central aggregator (or a competitor) might reconstruct their private data.
Privacy Target	**User-Level Privacy:** Protecting the identity/data of an individual person (e.g., through Differential Privacy).	**Silo/Institutional Privacy:** Protecting the entire organization's data assets and maintaining trade secrets.
Main Bottleneck	**Communication & Compute:** Devices have limited battery and bandwidth to run heavy encryption.	**Legal & Regulatory:** Security must be provable to auditors to satisfy laws like GDPR, HIPAA, or CCPA.
Trust Model	**Anonymous:** Participants are often unknown; defense relies on statistical robustness and noise.	**Known & Verified:** Participants are identified; defense relies on legal contracts and Secure Multi-Party Computation (SMPC).
Attack Surface	Backdoor attacks, label flipping, and membership inference.	Gradient leakage to competitors and model IP theft.

Table 10.1: Comparison of Security and Privacy Threats in FL

Bibliography

Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., and Zhang, L. (2016). Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318. ACM.

Bagdasaryan, E., Veit, A., Hua, Y., Estrin, D., and Shmatikov, V. (2020). How to backdoor federated learning. In *International Conference on Artificial Intelligence and Statistics*, pages 2938–2948. PMLR.

Biggio, B., Nelson, B., and Laskov, P. (2012). Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389*.

Blanchard, P., El Mhamdi, E. M., Guerraoui, R., and Stainer, J. (2017). Machine learning with adversaries: Byzantine tolerant gradient descent. *Advances in neural information processing systems*, 30.

Boenisch, F., Dziedzic, A., Schuster, R., Shamsabadi, A. S., Shumailov, I., and Papernot, N. (2023). When the curious abandon honesty: Federated learning is not private. In *2023 IEEE 8th European Symposium on Security and Privacy (EuroS&P)*, pages 175–199. IEEE.

Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., Ramage, D., Segal, A., and Seth, K. (2017). Practical secure aggregation for privacy-preserving machine learning. *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1175–1191.

Diffie, W. and Hellman, M. (1976). New directions in cryptography. *IEEE transactions on Information Theory*, 22(6):644–654.

Fang, M., Cao, X., Jia, J., and Gong, N. (2020). Local model poisoning attacks to byzantine-robust federated learning. In *29th USENIX Security Symposium*, pages 1605–1622.

Fowl, L., Geiping, J., Czaja, W., Goldblum, M., and Goldstein, T. (2022). Robbing the fed: Directly obtaining private data in federated learning with modified models.

Fredrikson, M., Jha, S., and Ristenpart, T. (2015). Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1322–1333.

Fung, C., Yoon, C. J., and Beschastnikh, I. (2020). The limitations of federated learning in sybil settings. In *23rd International Symposium on Research in Attacks, Intrusions and Defenses*, pages 301–316.

Geiping, J., Bauermeister, H., Dröge, H., and Moeller, M. (2020). Inverting gradients–how easy is it to break privacy in federated learning? *arXiv preprint arXiv:2003.14053*.

Geyer, R. C., Klein, T., and Nabi, M. (2017). Differentially private federated learning: A client level perspective. *arXiv preprint arXiv:1712.07557*.

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., et al. (2019). Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*.

Konečnỳ, J., McMahan, H. B., Ramage, D., and Richtárik, P. (2016). Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*.

Kuo, K., Yadav, C., and Smith, V. (2025). Research in collaborative learning does not serve cross-silo federated learning in practice.

Lamport, L., Shostak, R., and Pease, M. (1982). The byzantine generals problem. *ACM Trans. Program. Lang. Syst.*, 4(3):382–401.

McMahan, H. B., Moore, E., Ramage, D., Hampson, S., et al. (2016). Communication-efficient learning of deep networks from decentralized data. *arXiv preprint arXiv:1602.05629*.

McMahan, H. B., Ramage, D., Talwar, K., and Zhang, L. (2017). Learning differentially private recurrent language models. *arXiv preprint arXiv:1710.06963*.

Ramirez, M., Yoon, S., Damiani, E., Al Hamadi, H., Ardagna, C., Bena, N., Byon, Y.-J., Kim, T.-Y., Cho, C.-S., and Yeun, C. (2022). New data poison attacks on machine learning classifiers for mobile exfiltration.

Shamir, A. (1979). How to share a secret. *Communications of the ACM*, 22(11):612–613.

Shokri, R., Stronati, M., Song, C., and Shmatikov, V. (2017). Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18. IEEE.

Subramanyan, P., Sinha, R., Lebedev, I., Devadas, S., and Seshia, S. A. (2017). A formal foundation for secure remote execution of enclaves. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 2435–2450.

Tolpegin, V., Truex, S., Gursoy, M. E., and Liu, L. (2020). Data poisoning attacks against federated learning systems. In *European Symposium on Research in Computer Security*, pages 480–501.

Wang, H., Sreenivasan, K., Rajput, S., Vishwakarma, H., Agarwal, S., Sohn, J.-y., Lee, K., and Papailiopoulos, D. (2020). Attack of the tails: Yes, you really can backdoor federated learning. *Advances in Neural Information Processing Systems*, 33:16070–16084.

Xun, Y., Liang, S., Jia, X., Liu, X., and Cao, X. (2024). Ta-cleaner: A fine-grained text alignment backdoor defense strategy for multimodal contrastive learning.

Yao, A. C.-C. (1986). How to generate and exchange secrets. In *27th Annual Symposium on Foundations of Computer Science (sfcs 1986)*, pages 162–167. IEEE.

Yin, D., Chen, Y., Kannan, R., and Bartlett, P. (2018). Byzantine-robust distributed learning: Towards optimal statistical rates. In *International conference on machine learning*, pages 5650–5659. Pmlr.

Zhao, B., Mopuri, K. R., and Bilen, H. (2020). idlg: Improved deep leakage from gradients. *arXiv preprint arXiv:2001.02610*.

Zhao, J. C., Sharma, A., Elkordy, A. R., Ezzeldin, Y. H., Avestimehr, S., and Bagchi, S. (2024). Loki: Large-scale data reconstruction attack against federated learning through model manipulation. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 1287–1305. IEEE.

Zhu, L., Liu, Z., and Han, S. (2019). Deep leakage from gradients. *Advances in neural information processing systems*, 32.

Appendices

Appendix A

Federated Learning Threat Model and Notation

This appendix fixes notation and threat-model assumptions used throughout the FL-security appendix. Our goal is to make later chapters self-contained: each attack and defense can be read as operating under some combination of (i) what is observable to the adversary and (ii) what the adversary can control.

A.1 Federated learning setup (round-based training)

We consider a standard cross-device FL system with a central coordinator (server) and n clients. Training proceeds in rounds $t = 1, 2, \ldots$.

At round t the server selects a subset of clients $S^t \subseteq [n]$ (where $[n] = \{1, \ldots, n\}$), and sends the current global model $\mathbf{w}_G^{t-1}$ to each selected client.

Each selected client $i \in S^t$ performs local training on its private dataset $\mathcal{D}_i$ for E local steps/epochs and produces a local model $\mathbf{w}_i^t$ (or equivalently an update). We denote the local update as

$$\delta_i^t \stackrel{\text{def}}{=} \mathbf{w}_i^t - \mathbf{w}_G^{t-1}. \tag{A.1}$$

The server aggregates client updates using weights α_i (often proportional to local dataset sizes):

$$\mathbf{w}_G^t = \mathbf{w}_G^{t-1} + \sum_{i \in S^t} \alpha_i \delta_i^t. \tag{A.2}$$

A.2 Adversary roles and capabilities

We distinguish several adversary roles (which may be combined):

A.2.1 Malicious clients (Byzantine participants)

The attacker controls one or more FL clients and can:

- modify local training data (data poisoning),
- change the local training procedure,
- submit an arbitrary update δ_i^t (model poisoning).

A.2.2 Sybil attacker (many fake clients)

A Sybil attacker controls many accounts/devices, increasing its fraction of contributions in S^t. This amplifies both poisoning impact and the ability to overwhelm robust aggregation.

A.2.3 Curious or malicious server

The server typically observes client updates and can attempt to infer private client data from them. In some settings the server can also manipulate protocol details (client selection, model architecture) to increase leakage.

A.3 Observation regimes (what the adversary sees)

Many privacy attacks depend critically on what is observable. We classify attacks by the observation regime:

- **Per-client, per-round updates:** the adversary observes each δ_i^t individually.
- **Only aggregated updates:** the adversary sees only the aggregated update $\sum_{i \in S^t} \alpha_i \delta_i^t$ (or the resulting global model $\mathbf{w}_G^t$).
- **Multi-round observation:** the adversary observes many rounds, which can amplify leakage.

Secure aggregation (SA) as a defense. We do not assume SA is always enabled. Instead, later chapters list SA as a mitigation whenever it reduces observability of individual client updates (shifting from per-client to aggregate-only).

A.4 Security goals in FL

- **Integrity:** the learned model behaves correctly and does not contain backdoors.
- **Availability:** training converges and inference remains efficient (latency/energy).
- **Privacy:** private client data are not recoverable from updates or the trained model.

A.5 Defense families (where defenses operate)

- **Aggregation-time defenses:** robust aggregation, update clipping, anomaly detection.
- **Protocol-level defenses:** secure aggregation, authentication/anti-Sybil mechanisms.
- **Training-time privacy defenses:** differential privacy (clipping + noise) with utility trade-offs.
- **Verification-based defenses:** TEEs or verifiable computation (strong deployment assumptions).

Appendix B

Federated Learning Poisoning Attacks (Training-Time Integrity & Availability)

This appendix chapter summarizes *training-time* attacks against federated learning (FL), with an emphasis on **poisoning** attacks that aim to compromise the **integrity** (backdoors / targeted misclassification) or **availability** (prevent convergence, degrade utility) of the learned global model. We focus on the canonical cross-device FL setup where a central coordinator (server) aggregates client updates (e.g., FedAvg), and an attacker can control one or more participating clients.

Surveys and threat overviews include Bouacida and Mohapatra (2021); Xia et al. (2023). A useful perspective is to view poisoning as either (i) manipulation of training *data* at clients or (ii) manipulation of the *model update* (gradients/weights) contributed by clients.

B.1 Taxonomy of poisoning in federated learning

Data poisoning vs. model poisoning.

- **Data poisoning:** the attacker modifies the local training dataset at one or more clients (e.g., label flipping, backdoor triggers, or carefully crafted clean-label examples). Early classical poisoning work predates FL and includes label-flipping-style poisoning against SVMs (Biggio et al., 2012).

- **Model poisoning:** the attacker directly manipulates the client update (model parameters or gradients) sent to the server, potentially without modifying the local dataset.

Targeted vs. semi-targeted vs. untargeted. Following common categorization (e.g., Xia et al. (2023); Bhagoji et al. (2019)), poisoning goals include:

- **Targeted attacks:** enforce attacker-chosen behavior on specific inputs (often called backdoors).
- **Semi-targeted:** degrade performance on a subset of classes/tasks.
- **Untargeted attacks:** degrade overall utility or prevent convergence.

Clean-label vs. dirty-label data poisoning. In **dirty-label** poisoning the attacker can change both inputs and labels. In **clean-label** poisoning the attacker changes inputs but not labels (labels must pass verification); this constraint may arise in settings with label verification or curated pipelines (e.g., discussed in Doku and Rawat (2021)).

B.2 Attack/defense pairing I: Data poisoning in FL

Data poisoning in FL can be difficult to detect if the server does not have access to client data. At the same time, some defenses (robust losses, anomaly detection on data) require visibility into training examples, which may conflict with FL privacy goals.

B.2.1 Attacks: dirty-label and label-flipping/backdoor-style poisoning

Dirty-label poisoning can be highly effective in deep learning even with relatively few poisoned samples, because injected examples can shift the learned decision boundary (see discussion in Doku and Rawat (2021)). In FL, the impact of poisoned data is mediated through local training (number of epochs, batch size, optimizer) and through the aggregation rule.

B.2.2 Defenses: robust losses and data-based anomaly detection (often privacy-invasive)

Robust losses and anomaly detection over training data can detect label-flipping and some backdoor patterns, but they typically require access to the underlying examples or labels (at least for auditing), which may not be compatible with standard FL deployments.

B.3 Attack/defense pairing II: Model poisoning in FL

Model poisoning attacks manipulate the local update uploaded for aggregation. This is a powerful attack surface: even if raw client data are private, the server must receive *some* update, and an attacker who controls a client can choose this update arbitrarily unless there is strong verification.

B.3.1 Attacker goals and targeted objectives

Following Bhagoji et al. (2019), a targeted objective can be written as maximizing the number of attacker-chosen auxiliary inputs classified into attacker-chosen target labels :

$$\mathcal{A}(\mathcal{D}_m \cup \mathcal{D}_{\text{aux}}) = \max_{\mathbf{w}_G^t} \sum_{i=1}^{r} \mathbb{1}\big[f(\mathbf{x}_i; \mathbf{w}_G^t) = \tau_i\big], \tag{B.1}$$

where $\{(\mathbf{x}_i, y_i)\}_{i=1}^r$ is attacker-chosen auxiliary data with true labels y_i, and the attacker seeks predictions τ_i on these inputs. Untargeted poisoning instead aims to reduce general performance or prevent convergence.

B.3.2 Stealth: why poisoning must evade server-side checks

To succeed, an attacker often needs to be **stealthy**, i.e., avoid being filtered out by validation checks, outlier detection, or robust aggregation.

Two detection signals highlighted in Bhagoji et al. (2019) are:

(Defense signal) Validation-impact checks. One heuristic is to compare validation accuracy of a global model updated only by client i against a model updated by other clients. If client i causes a large drop, i is flagged. For example, a server may use a condition such as:

$$\sum_{\{x_j, y_j\} \in \mathcal{D}_{\text{test}}} \mathbb{1}[f(x_j; q_{D\setminus i}^t) = y_j] - \mathbb{1}[f(x_j; q_i^t) = y_j] < \gamma_t, \tag{B.2}$$

for threshold γ_t chosen by the server.

(Defense signal) Statistics of update vectors (distance/outlier scores). Pairwise distances between updates can indicate how different one update is from the rest; this is closely related to robust aggregation rules such as Krum Blanchard et al. (2017). One example statistic is to compute, for client m:

$$R_m = \Big[\min_{i \in [k]\setminus m} d(\delta_m^t, \delta_i^t),\ \max_{i \in [k]\setminus m} d(\delta_m^t, \delta_i^t)\Big], \tag{B.3}$$

and compare it to corresponding bounds from benign clients, requiring

$$\max\{|R_m^u - R_{\min,[k\setminus m]}^l|,\ |R_m^u - R_{\max,[k\setminus m]}^u|\} < \kappa_t, \tag{B.4}$$

for a server-chosen threshold κ_t.

B.3.3 Attack: optimization-based model poisoning with explicit boosting

A key difficulty for the attacker is that the exact current global model $\mathbf{w}_G^t$ may not be available; the attacker can only influence it through its submitted malicious update δ_m^t. Thus Bhagoji et al. (2019) formulates attacks using an estimate $\hat{\mathbf{w}}_G^t$ of the global model after aggregation:

$$\arg\min_{\mathbf{w}_m^t} L(\{x_i, \tau_i\}_{i=1}^r, \hat{\mathbf{w}}_G^t) \quad \text{s.t.} \quad \hat{\mathbf{w}}_G^t = g(\mathcal{I}_m^t), \tag{B.5}$$

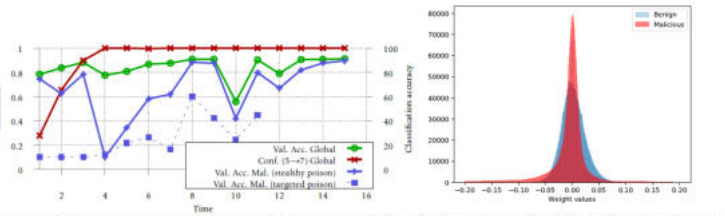
where $g(\cdot)$ is an estimator based on attacker information $\mathcal{I}_m^t$. A simple estimator is $\hat{\mathbf{w}}_G^t = \mathbf{w}_G^{t-1} + \alpha_m \delta_m^t$. This leads to an *explicit boosting factor* $\lambda = \frac{1}{\alpha_m}$.

Stealthy attack formulation. To increase stealth, Bhagoji et al. (2019) augments the adversarial objective with terms that (i) preserve main-task performance and (ii) keep the malicious update close to benign updates:

- main-task term: $L(\{x_i, y_i\}_{i=1}^{n_m}, \mathbf{w}_G^t)$,
- similarity term: $\rho \|\delta_m^t - \bar{\delta}_{\text{benign}}^{t-1}\|_2$,

where $\bar{\delta}_{\text{benign}}^{t-1} = \sum_{i \in [k] \setminus m} \alpha_i \delta_i^{t-1}$. A combined objective is:

$$\arg\min_{\mathbf{w}_m^t} \lambda L(\{x_i, \tau_i\}_{i=1}^r, \hat{\mathbf{w}}_G^t) + L(\{x_i, y_i\}_{i=1}^{n_m}, \mathbf{w}_G^t) + \rho \|\delta_m^t - \bar{\delta}_{\text{benign}}^{t-1}\|_2. \tag{B.6}$$

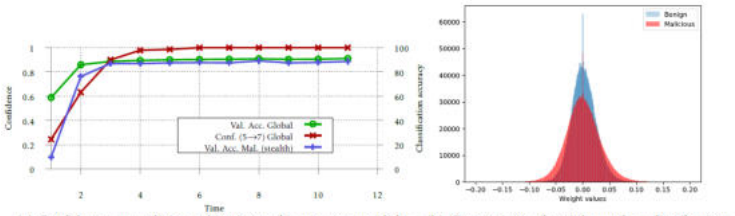


(a) Confidence on malicious objective and accuracy on validation data for $\mathbf{w}_G^t$. Stealth with respect to accuracy checking is also shown for both the stealthy and targeted model poisoning attacks.

(b) Comparison of weight update distributions for benign and malicious agents

Figure B.1: Stealthy model poisoning for CNN on FashionMNIST Bhagoji et al. (2019).

Attack refinement: alternating minimization. A practical strategy is to decouple optimization with respect to attack success and stealth. For each epoch, first optimize for the adversarial goal and boost the update to obtain a malicious model $\tilde{\mathbf{w}}_m^{i,t} = \mathbf{w}_m^{i-1,t} + \lambda \delta_m^{i,t}$, and then optimize it with respect to stealth constraints.



(a) Confidence on malicious objective and accuracy on validation data for $\mathbf{w}_G^t$. Stealth with respect to accuracy checking is also shown.

(b) Comparison of weight update distributions for benign and malicious agents

Figure B.2: Alternating minimization attack with distance constraints Bhagoji et al. (2019).

B.3.4 Defense: Byzantine-resilient aggregation (mostly for untargeted poisoning)

Let $\{\delta_i^t\}_{i=1}^n$ be client updates in round t.

- **Krum** (Blanchard et al., 2017): select an update whose sum of distances to its $n - f - 2$ nearest neighbor updates is minimal (assuming up to f Byzantine clients, with conditions such as $n \geq 2f + 3$).
- **Coordinate-wise median**(Pillutla et al., 2022): $\bar{\delta}^t = \text{coomed}(\{\delta_i^t\}_{i=1}^n)$.

Targeted model poisoning can remain effective against these defenses under adaptive attackers (Bhagoji et al., 2019).

B.3.5 Attack: fake clients and consistent steering (MPAF)

A simple baseline is injecting noise via fake clients (Cao and Gong, 2022):

- *random attack:* $g_i^t = -\lambda\epsilon$ for Gaussian noise ϵ,
- *history attack:* submit $-\lambda(\mathbf{w}^t - \mathbf{w}^{t-1})$.

These can be weak against defenses such as *trimmed mean* (Yin et al., 2018). MPAF (Cao and Gong, 2022) instead chooses a target model $\mathbf{w}'$ and drags the global model toward it:

$$\min_{\{g_i^t\}} \|\mathbf{w}^T - \mathbf{w}'\|, \tag{B.7}$$

implemented as:

$$g_i^t = \lambda(\mathbf{w}' - \mathbf{w}^t). \tag{B.8}$$

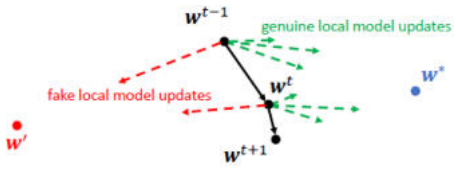


Figure B.3: Illustration of MPAF (Cao and Gong, 2022).

Common practical defenses: **Norm clipping** (Sun et al., 2019) bounds each update by M:

$$g \leftarrow \frac{g}{\max(1, \|g\|_2/M)}. \tag{B.9}$$

B.3.6 Attack: backdoors and model replacement

Model replacement attacks (Bagdasaryan et al., 2020) train a backdoored local model X and then scale the submitted update to "replace" the global model after averaging.

Let G^t denote the global model before aggregation. The attacker aims for:

$$X = G^t + \frac{\eta}{n} \sum_{i=1}^{m} (L_i^{t+1} - G^t), \tag{B.10}$$

and solves for a malicious local model $\tilde{L}_m^{t+1}$:

$$\tilde{L}_m^{t+1} = \frac{n}{\eta} X - \left(\frac{n}{\eta} - 1\right) G^t \approx \frac{n}{\eta}(X - G^t) + G^t. \tag{B.11}$$

Non-IID data and attacker selection frequency affect stealth and persistence; Bagdasaryan et al. (2020) discusses issues such as backdoor decay and strategies such as learning-rate control.

B.4 Verification-based defenses (stronger assumptions)

Some defenses attempt to verify client behavior or enforce honest execution:

- **Trusted execution environments (TEE):** Chen et al. (2020).
- **Proxy auditing / synthetic validation:** approaches that use proxy data or synthetic generation to audit updates Zhao et al. (2019), which may compromise privacy assumptions.

Appendix C

Gradient/Update Leakage Attacks in Federated Learning (Data Reconstruction)

This chapter summarizes **data reconstruction** attacks in FL, where an adversary attempts to recover private client training data from model updates (gradients or weight deltas). The key organizing principle is **what the adversary can observe**.

Throughout, let δ_i^t be client i's update in round t. Depending on the protocol and system configuration, an adversary may observe:

- **(I) Per-client, single-round updates:** observe one client's update δ_i^t for a single round.
- **(II) Aggregate-only updates:** observe only $\sum_{i \in S^t} \alpha_i \delta_i^t$ (or $\mathbf{w}_G^t$).
- **(III) Multi-round observation and malicious protocol design:** observe many rounds and/or manipulate the model to amplify leakage (e.g., insert layers that create analytical leakage).

We do not assume secure aggregation by default; instead, whenever SA prevents individual-update visibility we list it as a potential defense.

C.1 Observation regime I: Per-client, single-round updates

In this regime the adversary can access an individual client's update from a single round, e.g., a curious server without secure aggregation. Many canonical attacks are presented in this setting.

C.1.1 Single data-point (analytical leakage for simple architectures)

For single gradients a method is introduced in Aono et al. (2017b)(revised in Aono et al. (2017a)). The update in SGD training of NNs works in the following way ($\mathbf{w}$ parameter vector including bias vector $\mathbf{b}$):

$$\mathbf{w} = \mathbf{w} - \eta \nabla_{\mathbf{w}} f \tag{C.1}$$

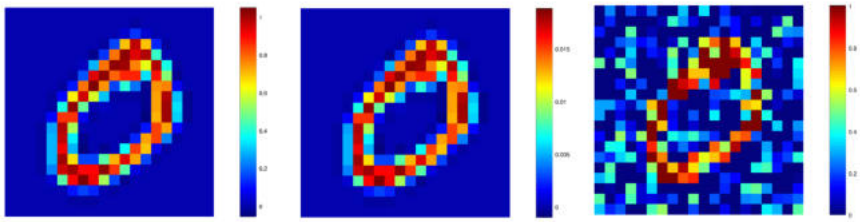
For a single neuron classifier with $\mathbf{w}_k$ denoting the weights for the kth input feature and b the bias, $f(\mathbf{W}, b, \mathbf{x}, y)$ denotes the training loss on one example, while $\phi(\cdot)$ is the neuron's activation function (and $\phi'(\cdot)$ its derivative). (illustrated for squared loss, but the "ratio" intuition carries to cross-entropy as well):

$$\Delta \mathbf{w}_k = \frac{\partial f(\mathbf{W}, b, \mathbf{x}, y)}{\partial \mathbf{w}_k} = 2(\phi(\mathbf{W}\mathbf{x} + b) - y)\phi'(\mathbf{W}\mathbf{x} + b) \cdot x_k, \tag{C.2}$$

$$\Delta b = \frac{\partial f(\mathbf{W}, b, \mathbf{x}, y)}{\partial b} = 2(\phi(\mathbf{W}\mathbf{x} + b) - y)\phi'(\mathbf{W}\mathbf{x} + b) \cdot 1. \tag{C.3}$$

Dividing gives the input coordinate:

$$x_k = \Delta \mathbf{w}_k / \Delta b. \tag{C.4}$$



(a) Original 20x20 image of handwritten number 0, seen as a vector over $\mathbb{R}^{400}$ fed to a neural network.

(b) Recovered image using 400/10285 (3.89%) gradients (see Sect.3, Example 2). The difference with the original (a) is only at the value bar.

(c) Recovered image using 400/10285 (3.89%) gradients (see Sect.3, Example 3). There are noises but the truth label 0 can still be seen.

Figure C.1: Reconstruction from single gradient in fully connected NN Aono et al. (2017b)

C.1.2 Deep Leakage from Gradients (DLG) and iDLG

Zhu et al. (2019) (and improved deep leakage, iDLG Zhao et al. (2020)) uses a different method: the attacker runs optimization on a synthetic input (initialized randomly) and adjusts it until the gradient induced by the synthetic input matches the observed gradient.

C.1.3 Arbitrary architecture for single datum

Denoting the loss of a network with weights $\mathbf{W}$ on dataset $(\mathbf{X}, \mathbf{y})$ as $f(\mathbf{W}, \mathbf{X}, \mathbf{y})$:

$$\nabla \mathbf{W} \stackrel{\text{def}}{=} \frac{\partial f(\mathbf{W}, \mathbf{X}, \mathbf{y})}{\partial \mathbf{W}}$$

To reconstruct a single data point $\mathbf{x}$, starting from random $\mathbf{x}'$ and random target $\mathbf{y}'$:
for $t = 1, \dots$:

1. $\nabla \mathbf{W}' \leftarrow \frac{\partial f(\mathbf{W}, \mathbf{x}', \mathbf{y}')}{\partial \mathbf{W}}$
2. $\Delta \leftarrow \|\nabla \mathbf{W}' - \nabla \mathbf{W}\|^2$
3. $\mathbf{x}' \leftarrow \mathbf{x}' - \eta \frac{\partial \Delta}{\partial \mathbf{x}'}$
4. $\mathbf{y}' \leftarrow \mathbf{y}' - \eta \frac{\partial \Delta}{\partial \mathbf{y}'}$

Label of single datum (iDLG)

The efficiency can be boosted by extracting the true label in a single step as proposed in iDLG (Zhao et al., 2020). When $\mathbf{y}$ is a one-hot vector, the sign structure of gradients can often reveal the correct class.

Classification is usually trained via cross-entropy loss, where for the correct class c, and the logits $\mathbf{y} = [y_1, \dots]$:

$$f(x, c) = -\log \frac{e^{y_c}}{\sum_j e^{y_j}}. \tag{C.5}$$

Thus

$$g_i \stackrel{\text{def}}{=} \frac{\partial f(x)}{\partial y_i} = \begin{cases} -1 + \frac{e^{y_i}}{\sum_j e^{y_j}} & < 0, \text{ if } i = c \\ \frac{e^{y_i}}{\sum_j e^{y_j}} & > 0, \text{ otherwise.} \end{cases} \tag{C.6}$$

Although gradients w.r.t. logits are not directly part of model updates, they can be inferred from the gradients of the last-layer weights. For the incoming weights of the ith output:

$$\nabla \mathbf{w}_i^{(L)} = \frac{\partial f(x, c)}{\partial \mathbf{w}_i^{(L)}} = g_i \cdot \mathbf{a}^{(L-1)}. \tag{C.7}$$

The correct class can be identified by picking the output whose incoming-weight gradient is directionally distinct:

$$c = i, \text{ iff } \nabla \mathbf{w}_i^{(L)^T} \cdot \nabla \mathbf{w}_j^{(L)} \leq 0, \ \forall j \neq i. \tag{C.8}$$

C.1.4 Reconstruction of a batch of data (single client batch)

DLG (Zhu et al., 2019) is also reported to work when the observed gradient corresponds to a mini-batch $\mathcal{B} = \{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_b, \mathbf{y}_b)\}$ drawn from a *single client*:

$$\nabla \mathbf{W} \stackrel{\text{def}}{=} \sum_{i=1}^{b} \frac{\partial f(\mathbf{W}, \mathbf{x_i}, \mathbf{y_i})}{\partial \mathbf{W}}. \tag{C.9}$$

A common heuristic is to update one synthetic datum at a time: for $t = 1, \ldots$:

1. $j = t \mod b$
2. $\nabla \mathbf{W}' \leftarrow \frac{\partial f(\mathbf{W}, \mathbf{x}', \mathbf{y}')}{\partial \mathbf{W}}$
3. $\Delta \leftarrow \|\nabla \mathbf{W}' - \nabla \mathbf{W}\|^2$
4. $\mathbf{x}'_\mathbf{j} \leftarrow \mathbf{x}'_\mathbf{j} - \eta \frac{\partial \Delta}{\partial \mathbf{x}'_\mathbf{j}}$
5. $\mathbf{y}'_\mathbf{j} \leftarrow \mathbf{y}'_\mathbf{j} - \eta \frac{\partial \Delta}{\partial \mathbf{y}'_\mathbf{j}}$

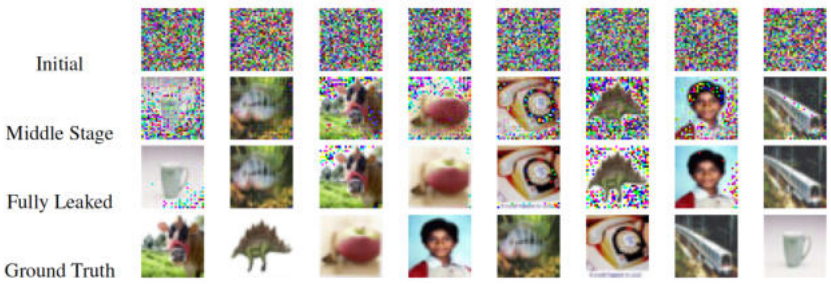


Figure C.2: Batch leakage Zhao et al. (2020)

C.1.5 Minimizing cosine distance (direction matching)

In Geiping et al. (2020) the authors analyze earlier methods and propose matching gradient *directions* (cosine distance), adding total variation (Rudin et al., 1992) ($TV(x)$) as an image prior:

$$d(x, y) = \frac{\langle x, y \rangle}{\|x\| \|y\|} \tag{C.10}$$

Objective:

$$\arg \min_{x \in [0,1]^p} 1 - \frac{\langle \nabla_\theta f_\theta(x, y), \nabla_\theta f_\theta(x^*, y) \rangle}{\|\nabla_\theta f_\theta(x, y)\| \|\nabla_\theta f_\theta(x^*, y)\|} + \alpha TV(x). \tag{C.11}$$

C.1.6 Defenses for regime I

When an adversary can observe per-client updates, defenses include:

- **Secure aggregation (SA):** prevents the server from observing individual client updates (shifts to regime II).
- **Differential privacy (DP):** clipping + noise can reduce inversion fidelity (with utility trade-offs).
- **Reduce exposed information:** avoid revealing per-layer gradients; restrict debug/telemetry exports.
- **Increase mixing:** larger batch sizes / more local averaging can make inversion harder, though not a complete defense.

C.2 Observation regime II: Aggregate-only updates

In this regime the adversary observes only an *aggregate* update $\sum_{i \in S^t} \alpha_i \delta_i^t$ (or the resulting $\mathbf{w}_G^t$). This is the natural visibility level under secure aggregation.

Why aggregate-only may still leak (and a concrete attack intuition). Even if individual updates are hidden, an attacker who sees aggregates over time may still be able to infer information about a target client when aggregates can be made "close" to single-client contributions. Two common mechanisms are:

- **Small cohorts:** if $|S^t|$ is small, each client has a large influence on the aggregate.
- **Differencing / isolation across rounds:** if the server (or a malicious scheduler) can select cohorts such that round t includes a target client and round $t+1$ includes the same cohort *without* that target, then subtracting the two aggregates approximately reveals the target's contribution (up to changes due to local randomness and model drift).

This highlights that "secure aggregation" is strongest when combined with robust cohort-formation rules and auditing of selection policies.

Defenses for regime II.

- enforce minimum cohort sizes and prevent per-user isolation,
- randomize and audit client selection policies (especially if the server is not fully trusted),
- apply DP at client or server level (depending on trust model),
- rate-limit participation / require mixing across many clients before any aggregate is revealed.

C.3 Observation regime III: Multi-round observation and malicious protocol design

Even if single-round leakage is limited, observing many rounds can strengthen reconstruction by providing many constraints. Moreover, a malicious or curious server may attempt to *shape* the model/training pipeline to amplify analytical leakage.

C.3.1 Linear-layer leakage attacks (analytical reconstruction perspective)

When a network starts with a fully connected layer, a single image x_i can be reconstructed if it activates neuron i:

$$x_i = \frac{\delta L}{\delta W_i} / \frac{\delta L}{\delta B_i}, \tag{C.12}$$

with $\frac{\delta L}{\delta W_i}$ denoting the weight gradient, and $\frac{\delta L}{\delta B_i}$ the bias gradient of the neuron.

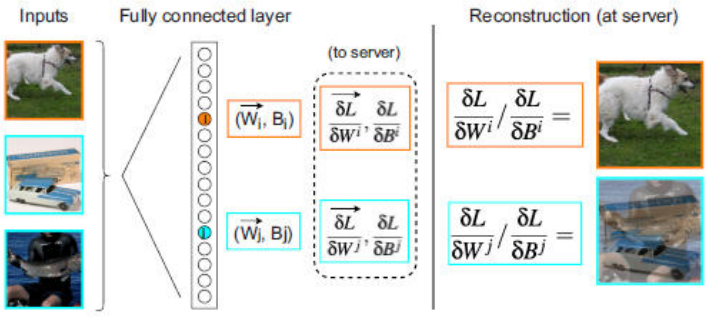


Figure C.3: Analytical FC leakage intuition. Source: Zhao et al. (2024)

C.3.2 Trap Weights

Trap weights (Boenisch et al., 2023) initialize weights to reduce activation probability. For each neuron:

1. initialize a randomly chosen half of the parameters from a Gaussian distribution;
2. assign to the second half of the weights the negative of the first ones;
3. scale down the positive weights:

$$W_{ij} \leftarrow W_{ij} \cdot c, \text{if } W_{ij} > 0, \quad c \in [0, 1] \tag{C.13}$$

C.3.3 Robbing the Fed

For image processing Fowl et al. (2022) proposes an *imprint module*: a fully connected layer implementing a binning method designed to route each input into a distinct "bin" (neuron). Biases are set as:

$$b_i = -\Phi^{-1}\left(\frac{i}{k}\right), \tag{C.14}$$

where k is the number of bins, and Φ^{-1} is the inverse of a cumulative distribution function of a statistic (e.g., brightness).

With $W_{ij} = \frac{1}{m}$ (input dimension m), a single image is expected to fall in a bin interval. Reconstruction can use differences between adjacent neurons:

$$x_j = \frac{\nabla_{W_{(i,j)}} L - \nabla_{W_{(i+1,j)}} L}{\nabla_{b_i} L - \nabla_{b_{i+1}} L}. \tag{C.15}$$

C.3.4 Loki

To overcome scalability limits of imprint-style leakage, Zhao et al. (2024) proposes an attack module consisting of a convolutional layer followed by two FC layers. The convolutional layer is used to push forward the image with identity-like filters, and different filter triplets are assigned to different clients, enabling reconstruction from client-specific parameter subsets.

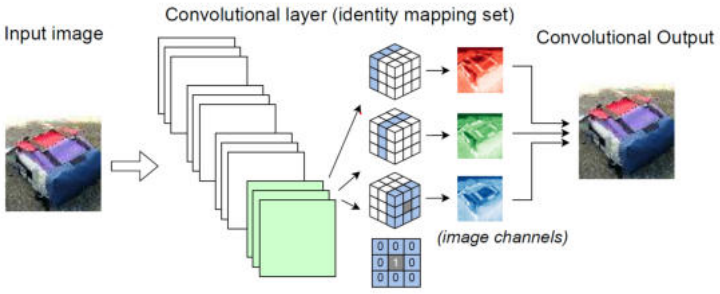


Figure C.4: Pushing the input forward with identity mapping in Loki Zhao et al. (2024).

C.3.5 Defenses for regime III

These attacks motivate defenses beyond "add noise":

- **Secure aggregation:** limits per-client visibility (helps, but does not prevent a malicious server from manipulating models).
- **Training-pipeline integrity controls:** audit and restrict server ability to introduce arbitrary layers (e.g., imprint modules).

- **Trusted execution / verifiable training (strong assumptions):** reduce server manipulation/observation capabilities.
- **Governance/engineering controls:** change management, signed architectures, client-side checks of expected model structure.

Appendix D

Inference on private data

Jayaraman et al. (2020) presents an analysis for inferring presence of records in the training data in realistic scenarios. The most important works that they have collected can be grouped into five group:

1. Membership inference Shokri et al. (2017a)Long et al. (2017)Salem et al. (2018)Yeom et al. (2018)
2. Attribute inference Fredrikson et al. (2014),Fredrikson et al. (2015)Yeom et al. (2018)
3. Property inference Ateniese et al. (2015)Ganju et al. (2018)
4. Model stealing Lowd and Meek (2005) Tramèr et al. (2016)
5. Hyperparameter stealing Wang and Gong (2018) Yan et al. (2020)

All these attacks are connected to the definition of differential privacy, since they build on differences in the work of models that either used ore not used specific data points in the training.

In general evaluation of the attacks are building on the privacy budget of the learning algorithms, thus application of deferentially private transformation can be understood as balancing between effectiveness of membership attacks, and usability of final model.

We found two important ideas for the attacks:

- The first general membership attack is presented in Shokri et al. (2017a). The attack trains shadow models that are similar to the attacked one, that is performing a similar task over a similar data-sets. These models are fed by different input data, and over their outputs a binary classifier attack model will be trained whose task is to decide whether the input as part of the training data or not. (Figure D.1)
- The method of Yeom et al. (2018) simplifies this method at a great extent. Omitting the shadow and attack model, the decision is made based on the expected value of confidence over the data points.

Membership inference In a Membership Inference Attack ($\texttt{Exp}(\texttt{Att}, A, n, \mathcal{D})$) (def in Yeom et al. (2018)) the adversary is given a data point $z = (x, y)$ and the task to decide, whether it has been included in the training data ($z \in S$) or not($z \in \mathcal{D} \setminus S$).

Formally the attack is a *membership experiment*. $\texttt{Exp.Att}$ is the attack, n is the size of training data S, and training algorithm A produces model $a = A(S)$. The attack proceeds as follows:

- Sample a training dataset $S \sim \mathcal{D}^n$.
- Choose $b \in \{0, 1\}$ uniformly at random.
- Draw a challenge point z:
  - $z \sim S$ if $b = 0$,
  - $z \sim \mathcal{D}$ if $b = 1$.
- If $\texttt{Att}(z, a, n, A, \mathcal{D}) = b$ return 1 (success), otherwise 0.

Membership advantage

$$\texttt{Adv}(\texttt{Att}, A, n, \mathcal{D}) = 2 \cdot \Pr(\texttt{Exp}(\texttt{Att}, A, n, \mathcal{D}) = 1) - 1, \tag{D.1}$$

which ranges from 0 (random guessing) to 1 (perfect inference).

Jayaraman and Evans (2019) presents a comparison of performance of different MIAs under various DP mechanisms, while Bernau et al. (2019) gives a theoretical upper bound on the success an adversary can achieve in MIA.

Uncertainties in performance evaluation

As Humphries et al. (2020) points out, in many cases the guarantees and empirical performance measurements are not sufficient. Theoretical upper bounds and empirical/theoretical lower bounds often rely on simplifying assumptions:

- In this scheme z drawn from $\mathcal{D}$ can also be part of S. Other works use $z \sim \mathcal{D} \setminus S$ for $b = 1$ Shokri et al. (2017b).
- Assumption of independence of records, which can be unrealistic; dependencies can amplify leakage Liu et al. (2016); Almadhoun et al. (2020); Tschantz et al. (2020).
- Jayaraman et al. (2020) considers more realistic unbalanced cases: $\Pr(b = 0) \neq \Pr(b = 1)$.
- The attacker may or may not have access to the same data distribution.

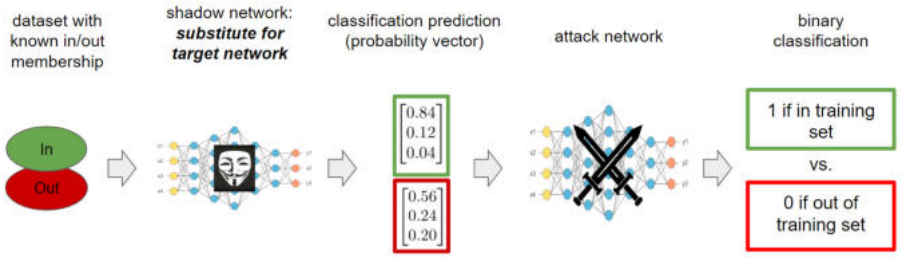


Figure D.1: Membership inference with shadow models Shokri et al. (2017a)[1]

Missing attributes

The essence of attribute inference Fredrikson et al. (2014); Yeom et al. (2018) is to use the model distribution to infer the most probable values of missing attributes $\mathbf{x}'$, conditioned on known attributes $\mathbf{x}$:

$$\arg\max_{\mathbf{x}'} \Pr_m(\mathbf{x}'|\mathbf{x}) = \arg\min_{\mathbf{x}',\mathbf{y}} f((\mathbf{x}', \mathbf{x}), \mathbf{y}). \tag{D.2}$$

This method can be more powerful if we already know that a person belongs to the training data. Intuitively, since the loss has been minimized on the given example, the attribute values that minimize the loss are more likely to be the true values.

Specifically in medicine, one application in Fredrikson et al. (2014) shows that one could identify genetic markers based on warfarin dosage output. Building on white-box information Wu et al. (2016), more challenging attacks are possible, such as recovering faces from training data Fredrikson et al. (2015).

Result

Successful MIAs might lead to leakage of sensitive data.

Defense

A standard countermeasure is blurring out the contribution of individual records with *differentially private* transformations during training.

Summary

These attacks can reveal whether a person was included in a training dataset. Combined with external knowledge or data catalogs, this may enable sensitive inferences.

Bibliography

Almadhoun, N., Ayday, E., and Ulusoy, Ö. (2020). Inference attacks against differentially private query results from genomic datasets including dependent tuples. *Bioinformatics*, 36(Supplement_1):i136–i145.

Aono, Y., Hayashi, T., Wang, L., Moriai, S., et al. (2017a). Privacy-preserving deep learning: Revisited and enhanced. In *International Conference on Applications and Techniques in Information Security*, pages 100–110. Springer.

Aono, Y., Hayashi, T., Wang, L., Moriai, S., et al. (2017b). Privacy-preserving deep learning via additively homomorphic encryption. *IEEE Transactions on Information Forensics and Security*, 13(5):1333–1345.

Ateniese, G., Mancini, L. V., Spognardi, A., Villani, A., Vitali, D., and Felici, G. (2015). Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers. *International Journal of Security and Networks*, 10(3):137–150.

Bagdasaryan, E., Veit, A., Hua, Y., Estrin, D., and Shmatikov, V. (2020). How to backdoor federated learning. In *International Conference on Artificial Intelligence and Statistics*, pages 2938–2948. PMLR.

Bernau, D., Grassal, P.-W., Robl, J., and Kerschbaum, F. (2019). Assessing differentially private deep learning with membership inference. *arXiv preprint arXiv:1912.11328*.

Bhagoji, A. N., Chakraborty, S., Mittal, P., and Calo, S. (2019). Analyzing federated learning through an adversarial lens. In *International conference on machine learning*, pages 634–643. PMLR.

Biggio, B., Nelson, B., and Laskov, P. (2012). Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389*.

Blanchard, P., El Mhamdi, E. M., Guerraoui, R., and Stainer, J. (2017). Machine learning with adversaries: Byzantine tolerant gradient descent. *Advances in neural information processing systems*, 30.

Boenisch, F., Dziedzic, A., Schuster, R., Shamsabadi, A. S., Shumailov, I., and Papernot, N. (2023). When the curious abandon honesty: Federated learning is not private. In *2023 IEEE 8th European Symposium on Security and Privacy (EuroS&P)*, pages 175–199. IEEE.

Bouacida, N. and Mohapatra, P. (2021). Vulnerabilities in federated learning. *IEEe Access*, 9:63229–63249.

Cao, X. and Gong, N. Z. (2022). Mpaf: Model poisoning attacks to federated learning based on fake clients. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3396–3404.

Chen, Y., Luo, F., Li, T., Xiang, T., Liu, Z., and Li, J. (2020). A training-integrity privacy-preserving federated learning scheme with trusted execution environment. *Information Sciences*, 522:69–79.

Doku, R. and Rawat, D. B. (2021). Mitigating data poisoning attacks on a federated learning-edge computing network. In *2021 IEEE 18th Annual Consumer Communications Networking Conference (CCNC)*, pages 1–6.

Fowl, L., Geiping, J., Czaja, W., Goldblum, M., and Goldstein, T. (2022). Robbing the fed: Directly obtaining private data in federated learning with modified models.

Fredrikson, M., Jha, S., and Ristenpart, T. (2015). Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1322–1333.

Fredrikson, M., Lantz, E., Jha, S., Lin, S., Page, D., and Ristenpart, T. (2014). Privacy in pharmacogenetics: An end-to-end case study of personalized warfarin dosing. In *23rd {USENIX} Security Symposium ({USENIX} Security 14)*, pages 17–32.

Ganju, K., Wang, Q., Yang, W., Gunter, C. A., and Borisov, N. (2018). Property inference attacks on fully connected neural networks using permutation invariant representations. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 619–633.

Geiping, J., Bauermeister, H., Dröge, H., and Moeller, M. (2020). Inverting gradients–how easy is it to break privacy in federated learning? *arXiv preprint arXiv:2003.14053*.

Humphries, T., Rafuse, M., Tulloch, L., Oya, S., Goldberg, I., Hengartner, U., and Kerschbaum, F. (2020). Differentially private learning does not bound membership inference. *arXiv preprint arXiv:2010.12112*.

Jayaraman, B. and Evans, D. (2019). Evaluating differentially private machine learning in practice. In *28th {USENIX} Security Symposium ({USENIX} Security 19)*, pages 1895–1912.

Jayaraman, B., Wang, L., Knipmeyer, K., Gu, Q., and Evans, D. (2020). Revisiting membership inference under realistic assumptions. *arXiv preprint arXiv:2005.10881*.

Liu, C., Chakraborty, S., and Mittal, P. (2016). Dependence makes you vulnberable: Differential privacy under dependent tuples. In *NDSS*, volume 16, pages 21–24.

Long, Y., Bindschaedler, V., and Gunter, C. A. (2017). Towards measuring membership privacy. *arXiv preprint arXiv:1712.09136*.

Lowd, D. and Meek, C. (2005). Adversarial learning. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 641–647.

Pillutla, K., Kakade, S. M., and Harchaoui, Z. (2022). Robust aggregation for federated learning. *IEEE Transactions on Signal Processing*, 70:1142–1154.

Rudin, L. I., Osher, S., and Fatemi, E. (1992). Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4):259–268.

Salem, A., Zhang, Y., Humbert, M., Berrang, P., Fritz, M., and Backes, M. (2018). Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models. *arXiv preprint arXiv:1806.01246*.

Shokri, R., Stronati, M., Song, C., and Shmatikov, V. (2017a). Membership inference attacks against machine learning models.

Shokri, R., Stronati, M., Song, C., and Shmatikov, V. (2017b). Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18. IEEE.

Sun, Z., Kairouz, P., Suresh, A. T., and McMahan, H. B. (2019). Can you really backdoor federated learning? *arXiv preprint arXiv:1911.07963*.

Tramèr, F., Zhang, F., Juels, A., Reiter, M. K., and Ristenpart, T. (2016). Stealing machine learning models via prediction apis. In *25th {USENIX} Security Symposium ({USENIX} Security 16)*, pages 601–618.

Tschantz, M. C., Sen, S., and Datta, A. (2020). Sok: Differential privacy as a causal property. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 354–371. IEEE.

Wang, B. and Gong, N. Z. (2018). Stealing hyperparameters in machine learning. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 36–52. IEEE.

Wu, X., Fredrikson, M., Jha, S., and Naughton, J. F. (2016). A methodology for formalizing model-inversion attacks. In *2016 IEEE 29th Computer Security Foundations Symposium (CSF)*, pages 355–370. IEEE.

Xia, G., Chen, J., Yu, C., and Ma, J. (2023). Poisoning attacks in federated learning: A survey. *Ieee Access*, 11:10708–10722.

Yan, M., Fletcher, C. W., and Torrellas, J. (2020). Cache telepathy: Leveraging shared resource attacks to learn {DNN} architectures. In *29th {USENIX} Security Symposium ({USENIX} Security 20)*, pages 2003–2020.

Yeom, S., Giacomelli, I., Fredrikson, M., and Jha, S. (2018). Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, pages 268–282. IEEE.

Yin, D., Chen, Y., Kannan, R., and Bartlett, P. (2018). Byzantine-robust distributed learning: Towards optimal statistical rates. In *International conference on machine learning*, pages 5650–5659. Pmlr.

Zhao, B., Mopuri, K. R., and Bilen, H. (2020). idlg: Improved deep leakage from gradients. *arXiv preprint arXiv:2001.02610*.

Zhao, J. C., Sharma, A., Elkordy, A. R., Ezzeldin, Y. H., Avestimehr, S., and Bagchi, S. (2024). Loki: Large-scale data reconstruction attack against federated learning through model manipulation. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 1287–1305. IEEE.

Zhao, Y., Chen, J., Zhang, J., Wu, D., Teng, J., and Yu, S. (2019). Pdgan: A novel poisoning defense method in federated learning using generative adversarial network. In *International conference on algorithms and architectures for parallel processing*, pages 595–609. Springer.

Zhu, L., Liu, Z., and Han, S. (2019). Deep leakage from gradients. *Advances in neural information processing systems*, 32.

Eötvös Loránd University (ELTE)
Faculty of Informatics (IK)
Pázmány Péter sétány 1/c
1117 Budapest, Hungary

PHYSICAL AND PERSONNEL SECURITY

Introduction to Data Security

Imre Lendák, PhD, GICSP

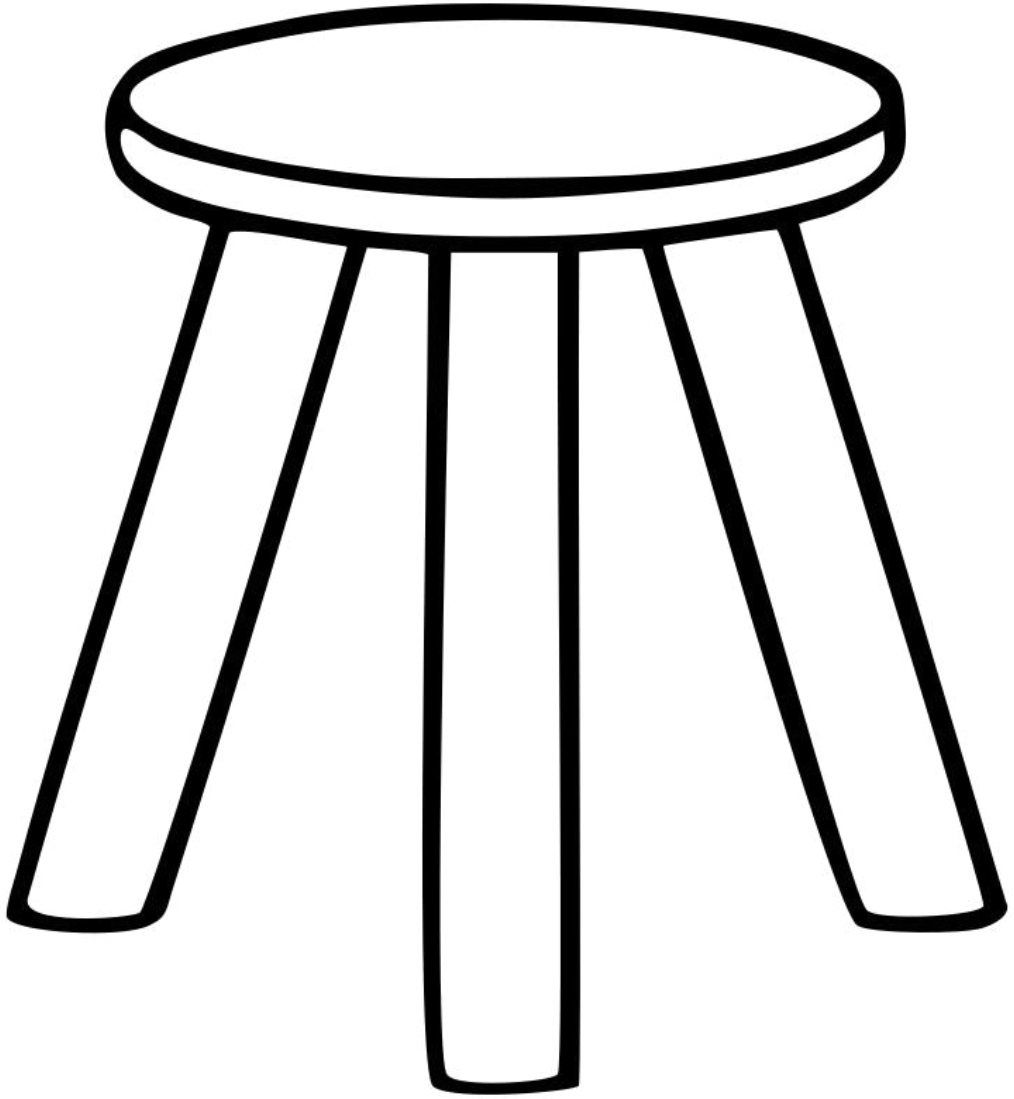
2026
Budapest, Hungary

Presentation outline

- Basic definition and data lifecycle
- Attackers and attack types
- Methods
- Physical security
  - Outside the facility
  - Physical access points
  - Inside the facility
  - Inside the control room
- Personnel security in brief

The CIA triad & CIA+AN

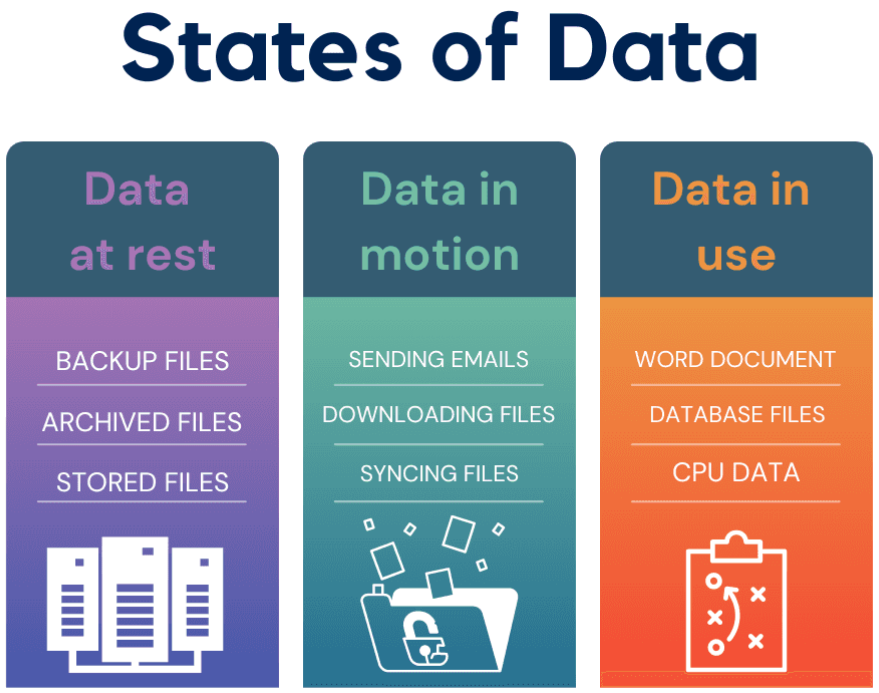
- **Confidentiality** = Authorized entities can access data & services
- **Integrity** = Authorized entities modify system configuration
  - **Discuss:** What is configuration in different contexts?
- **Availability** = Data and services are accessible within predefined time constraints
- +**Authentication** = Verify the identity of a user, process, or device
- +**Non-repudiation** = 3rd-party validation of message integrity and origin (from specific entity with key)



Additional key definitions

- Definitions based on the NIST Glossary
- **Cybersecurity** = Damage prevention, protection and restoration of information infrastructure and data to ensure its confidentiality, integrity, availability, authentication, and nonrepudiation.
- **Vulnerability** = Weakness in an information system (system security procedures, internal controls, or implementation) that could be exploited or triggered by a threat source.
  - TTP = Tools, Technology and Process
- **Threat** = Any circumstance or event with the potential to adversely impact an information system via unauthorized access, destruction, disclosure, modification of information, and/or denial of service → Potential negative impact on the CIA of an information system.
- **Threat actor** = An individual or a group posing a threat.
- **Attack** = Malicious activity which attempts to collect, disrupt, deny, degrade, or destroy information system resources or the information itself → Activity which negatively impacts CIA.
- **Risk** = A measure of the extent to which an entity is threatened by a potential circumstance or event, and typically a function of likelihood and impact.
- **Security control** = A safeguard or countermeasure prescribed for an information system, or an organization designed to protect information and meet security requirements.

Data in…



https://estuary.dev/data-in-motion/

- Data at rest
  - Storage (electronic, paper)
  - Cloud
  - Backup (we really need this!)
  - **Discussion:** anywhere else?
- Data in transit
  - Data communicated over critical information infrastructures
- Data in use
  - Data is loaded into RAM
  - Calculations are performed in a single or multiple physical or virtual computers (or containers)

ATTACKERS AND ATTACK TYPES

Adversaries

Adversary	Description (as defined in NIST IR 7628)
Nation states	State-run, well organized and financed. Use foreign service agents to gather classified or critical information from countries viewed as hostile or as having an economic, military or a political advantage.
Hackers	A group of individuals (e.g., hackers, phreakers, crackers, trashers, and pirates) who attack networks and systems seeking to exploit the vulnerabilities in operating systems or other flaws.
Terrorists / Cyberterrorists	Individuals or groups operating domestically or internationally who represent various terrorist or extremist groups that use violence or the threat of violence to incite fear with the intention of coercing or intimidating governments or societies into succumbing to their demands.
Organized crime	Coordinated criminal activities including cyber extortion, IP theft and other. Organized and well-financed criminal organization.
Other criminal elements	Other facets of the criminal community, which is normally not well organized or financed. Normally consists of a few individuals, or of one individual acting alone.
Industrial competitors	Foreign and domestic corporations operating in a competitive market and often engaged in the illegal gathering of information from competitors or foreign governments in the form of corporate espionage.
Disgruntled employees	Angry, dissatisfied individuals with the potential to inflict harm on critical infrastructures. This can represent an insider threat depending on the current state of the individual's employment and access to the systems.
Careless or poorly trained employees*	Those users who, either through lack of training, lack of concern, or lack of attentiveness pose a threat to a CI. This is another example of an insider threat or adversary.

Cyber attacks targeting individuals (2025)

- **Spam:** Unwanted email. Often contains malicious links or attachments. Countermeasures: anti-malware.
- **Online scams:** Nigerian prince and its novel variants. Countermeasures: check message source, follow trends.
- **Hijacking of electronic accounts:** Weak password or other vuln is used to hijack account. Countermeasures: multi-factor auth (MFA).
- **Ransomware:** Vuln is used to obtain access, encrypt files and request ransom for decode key. Countermeasure: training, backup.
- **Discussion:** Other?

Cyber attacks in business envs (2025)

- **Ransomware:** The same as against individuals, but potentially higher impact → higher ransom. Examples: UNIX, healthcare providers, Maersk.
- **Business account compromise:** Hijack and misuse business account(s) (e.g., send spam) → negative impact on business.
  - **Discuss:** Which?
- **Data breaches:** Trick employee to send data or exfiltrate after system breach.
  - **Discuss:** Often combined with ransomware. How?
- **Internet scams:**
  - **CEO fraud:** Attacker claims to be the CEO and instructs employee to perform unwanted action.
    - **Discuss:** Since ~2024 the use of deepfakes (audio, video) further aggravates this threat.
  - **Unwanted invoice actions:** Attackers inject unwanted invoices into legitimate exchanges and trick employees into transmitting funds to their accounts.
    - **Discuss:** Anybody heard about such attacks?
- **Supply chain attacks:** Adversary hacks a solution or hardware provider, introduces a vulnerability into a product (backdoor) which is subsequently delivered to a large public service organization or multinational company, allowing the adversary to gain the initial foothold easier
  - Examples: Solarwinds 2020, Kaseya 2021, Hezbollah pager attack 2024

ATTACK & DEFENSE METHODS

Cyber Kill Chain model

1. Reconnaissance
2. Weaponization
3. Delivery
4. Exploitation
5. Installation
6. Command & Control
7. Actions on Objectives

Stuxnet in a nutshell

- **Date:** June 2010 (detection)
- **Goal:** destroy nuclear fuel enrichment centrifuges via cyber attack
- **Motive:** hinder nuclear program via centrifuge destruction
- **Method:** cyber weapon → worm ~500kB, zero-day exploints, hard-coded passwords
- **Source:** (maybe) USA & Israel (?)
- **Outcome:** official, well-publicized start of global cyberwarfare

Stuxnet (2010) and the kill chain

Reconnaissance	• Geo location; physical security; equipment; SCADA
Weaponization	• Develop complex malware; steal digital certs (to hide malware)
Delivery	• USB drive (most probably)
Exploitation	• Windows zero days; hardcoded passwords
Installation	• Lateral movement; malware deployment on maintenance and operations workstations
Command & Control	• PLC recipe modification; hide traces
Actions on Objectives	• Send commands to destroy centrifuges

ISO 27001 CONTROLS

Physical Controls

ANNEX A OF ISO 27001:2022

The physical control theme includes 14 controls aimed at protecting an organisation's physical environment (such as buildings, offices, data centres, and equipment) from unauthorised access, damage, or interference.

7.1 Physical security perimeters

Security perimeters shall be defined and used to protect areas that contain information and other associated assets.

7.2 Physical entry

Secure areas shall be protected by appropriate entry controls and access points.

7.3 Securing offices, rooms and facilities

Security perimeters shall be defined and used to protect areas that contain information and other associated assets.

7.4 Physical security monitoring

Premises shall be continuously monitored for unauthorised physical access.

7.5 Protecting against physical and environmental threats

Protection against physical and environmental threats, such as natural disasters and other intentional or unintentional physical threats to infrastructure shall be designed and implemented.

7.6 Working in secure areas

Security measures for working in secure areas shall be designed and implemented.

7.6 Clear desk and clear screen

Clear desk rules for papers and removable storage media and clear screen rules for information processing facilities shall be defined and appropriately enforced.

7.8 Equipment siting and protection

Equipment shall be sited securely and protected.

7.9 Security of assets off-premises

Off-site assets shall be protected.

7.10 Storage media

Storage media shall be managed through their life cycle of acquisition, use, transportation and disposal in accordance with the organisation's classification scheme and handling requirements.

7.11 Supporting utilities

Information processing facilities shall be protected from power failures and other disruptions caused by failures in supporting utilities.

7.12 Cabling security

Cables carrying power, data or supporting information services shall be protected from interception, interference or damage.

7.13 Equipment maintenance

Equipment shall be maintained correctly to ensure availability, integrity and confidentiality of information.

7.14 Secure disposal or re-use of equipment

Items of equipment containing storage media shall be verified to ensure that any sensitive data and licensed software has been removed or securely overwritten prior to disposal or re-use.

Physical and personnel security

OUTSIDE THE FACILITY

The ‘onion’ of physical security

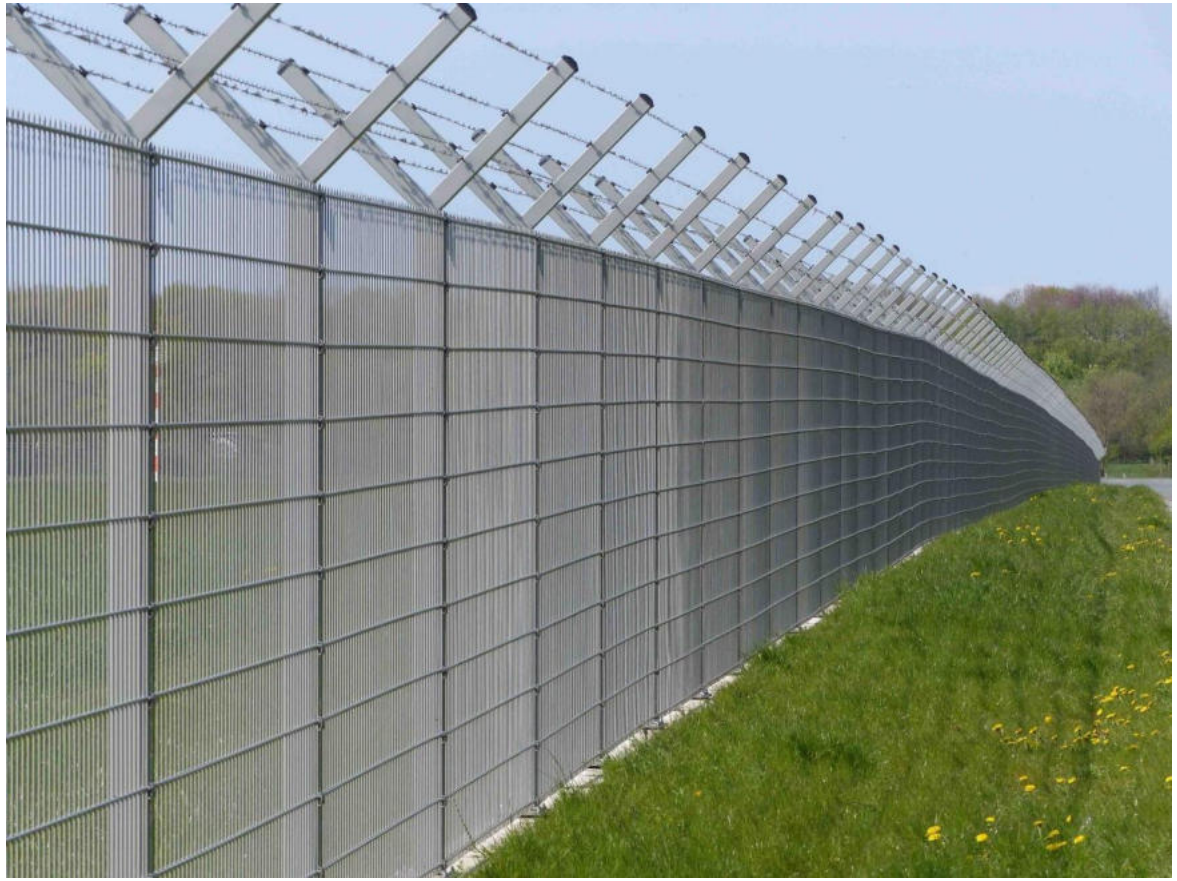
- Well-chosen location of the industrial/infrastructure facility
- Cleared area around the location
- Fence with vehicle and personnel access points – optionally with bomb detection
- On-premise surveillance system with cameras, sensors and guards&dogs
- Secured external walls and windows
- Secured doors with multi-factor authentication between internal security zones
- Secured access to IT infrastructure e.g., server rooms
- Secured access to sensitive IT & OT equipment e.g., rack protection, locked equipment cabinets with PLCs
- Physical port security
- (Optional) Redundant utilities i.e., electricity, water, communications, transport.

Location

- Avoid locations prone no natural disasters e.g., tornadoes, floods, earthquake fault lines → tsunamis
- Choose neighbors wisely → avoid prisons, airports and chemical plants which might cause unplanned disturbances
- Cleared ground around the industrial/infrastructure facility
  - Clear overgrown trees and other large obstacles to visibility and access, e.g. large boulders
  - ~30m cleared buffer zone around the facility to improve the efficiency of the camera system and guards

Fences

- Fences are the first line of defense
  - The entire fence can be regarded as a potential physical access point (PAP)
- 2-meter-high fences deter casual intruders from trespassing
- 2.5-meter and higher fences will stop even the most determined intruders



https://te-fence.com/high-security-fences/

Vehicle and personnel entry + bomb detection

- Vehicle and personnel entry points are physical access points which allow policy-allowed physical access to the ICS facility
- Single main entrance + back entrance for delivery & shipping
- ‘Staffed’ guard station
- Retractable ram posts
- Fire doors exit only
- Entrance with badge/card swipe or PIN code + multi-factor authentication
- Strict visiting policies for guests and contractors
- (Optional) Highly sensitive, high-risk facilities need to implement manual (gate guards with mirrors to peek underneath vehicles entering the compound) or assisted bomb detection

Physical barriers

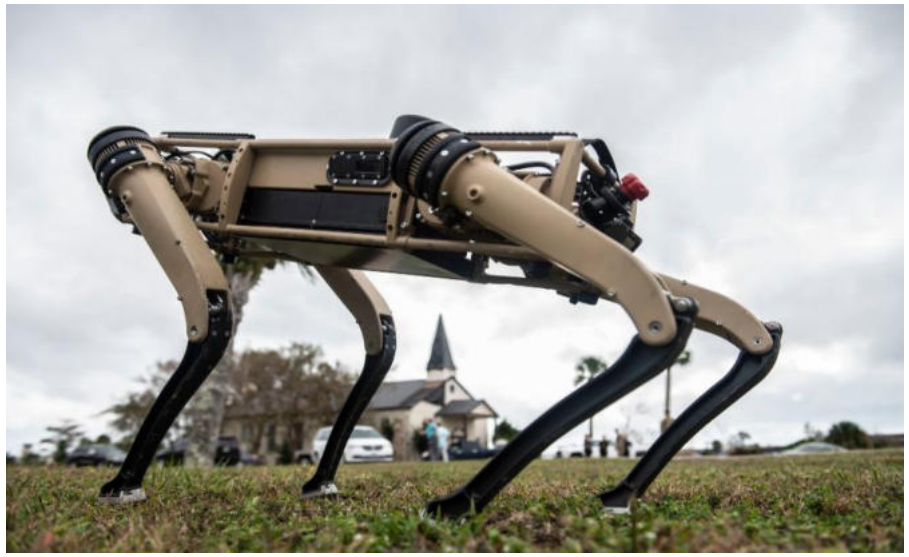
Armed guards

- Civilian clothing
- Security training is a plus (military, national guard, police)

- Which infrastructures do need physical guards → where to station them?

https://securityguardsuk.com/news/rules-security-guard-must-obey/

Dogs



- Traditionally dogs were used on physical perimeters → augment the human guards
- Living dogs might be soon partially substituted by their robotic brethren

https://www.securitymagazine.com/articles/93941-us-air-force-implements-dog-like-robots-into-patrolling-regiment-to-enhance-security-and-safety

Physical and personnel security

PHYSICAL ACCESS POINTS

Walls

- 30 cm or thicker external concrete walls provide adequate protection against severe weather, intruders and explosive devices
- Additional protection can be attained by using Kevlar
- Interior walls need to go all the way to the ceiling → do not allow intruders to crawl over in case of a dropped ceiling

https://www.indiamart.com/proddetail/precast-concrete-security-wall-22189506888.html

Doors & security portals

- Secure doors
  - Made from metal or use metal structures
  - Strong locks
  - Hinges on the more secure side
- Security portals
  - One person at a time → no piggy-backing

Rooftops

- Traditionally the focus was on safety → nobody falls off the roof
- Latest challenges
  - Drone attacks
  - Breaking and entering via roofs
  - Protecting the equipment usually mounted on rooftops → which?
- Consider basements as well (!)

Windows

https://mullerexteriors.com/better-window-safety-5-ways-to-secure-your-windows-against-burglars/

- Avoid windows as much as possible as they are the weakest elements of exterior defense
  - Exception: offices or rooms where employees take breaks
- Glass types to consider
  - Heat strengthened glass
  - Fully tempered glass
  - Heat-soaked tempered glass
  - Laminated glass
  - Wire glass

On-premise surveillance

External

- DEF: External surveillance is implemented on the outer physical perimeter of the ICS facility
- IP cameras or closed-circuit TV
- Security guards & dogs
- Use a clever combination of
  - Motion detection devices
  - Low-light cameras
  - Pan-tilt-zoom cameras
  - Fixed cameras
- Store sensor recordings offsite for planned amount of time

Internal

- DEF: Internal surveillances is implemented inside the facility
- Cover every physical access point
- Watch the entrances and exits both outside and inside, especially at the more sensitive (physical) zones
- Keep track of who was where and when → spot when equipment disappears

Sending and destroying data stores

Send data

- Might need to send data to external entities on CD/DVD, USB, paper
- Encrypt the data
- Do not mark the envelopes
- Choose postal services with electronic tracking
- Proof of receipt

Data wipes

- Always attempt to counter dumpster diving (!)
- Use a shredder to destroy paper
- Physically break CD/DVD/Blue Ray disks
- Magnetically erase and physically destroy magnetic drives
- Dumpster bins and containers keep locked whenever possible
- Consider subcontracting a data wipe specialist company

Physical and personnel security

INSIDE THE FACILITY

Computing equipment threats

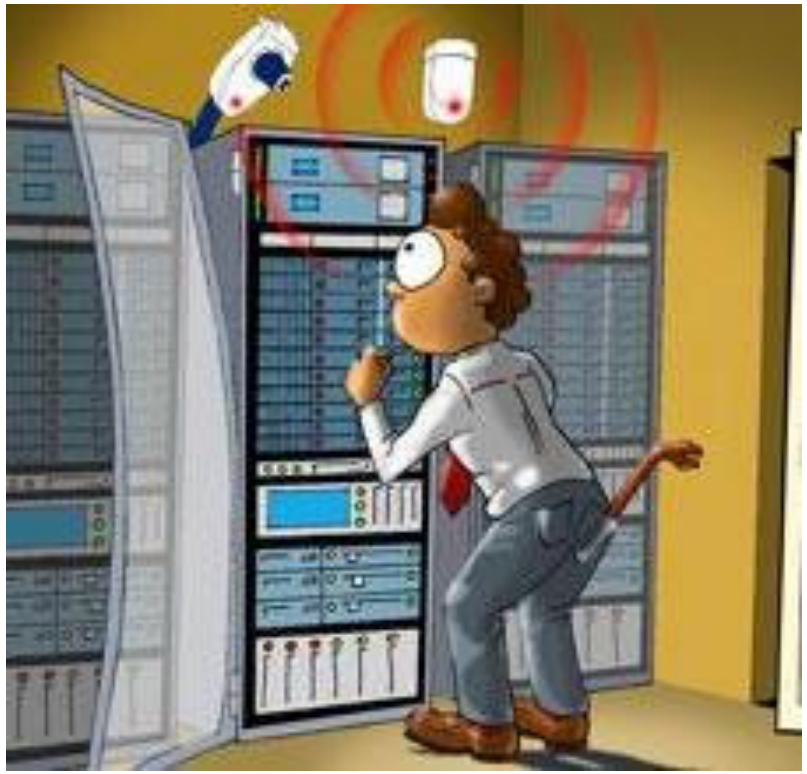
- Natural events (e.g., floods, earthquakes, and tornados)
- Other environmental conditions (e.g., extreme temperatures, high humidity, heavy rains, and lightning)
- Intentional acts of destruction (e.g., theft, vandalism, and arson)
- Unintentionally destructive acts (e.g., spilled drinks, overloaded electrical outlets, and bad plumbing)

https://nces.ed.gov/pubs98/safetech/chapter5.asp

Secure access to the process environment

- Consider the process environment as a high security area and allow access via a limited number of doors
- Use strict (multi-factor) authentication e.g., swipe cards and/or biometric reading devices
- Audit each entry and exit into and out of the process/production area
- Representatives of vendors and visitors must be accompanied by an employee during 100% of their visits

Secure access to IT infrastructure



- Server rooms behind single, locked doors
  - Multi-factor authentication, e.g., biometric sensor + card
  - Strictly log physical access
- Shielded and hidden cabling (network, electricity)
- Fire protection
- Secured suspended ceiling if cabling is above
  - Similar for raised floors (!)
- Video surveillance

Data center secure storage

- Data centers might host customer-owned racks
- Physical fences
- Secure doors
- Secure rooftops
- Well-hidden cabling (electricity & network)

https://www.storagesystems.ie/products/details/data-center-cages/76

Structural lowered ceilings

- Hidden or in plain sight
- Allow on-site administrators to implement structural cabling
- Cabling should be hidden outside server rooms

Raised floors

- Structured cabling under the floor
- Often easier to access and hidden from plain sight (when compared to lowered ceilings)

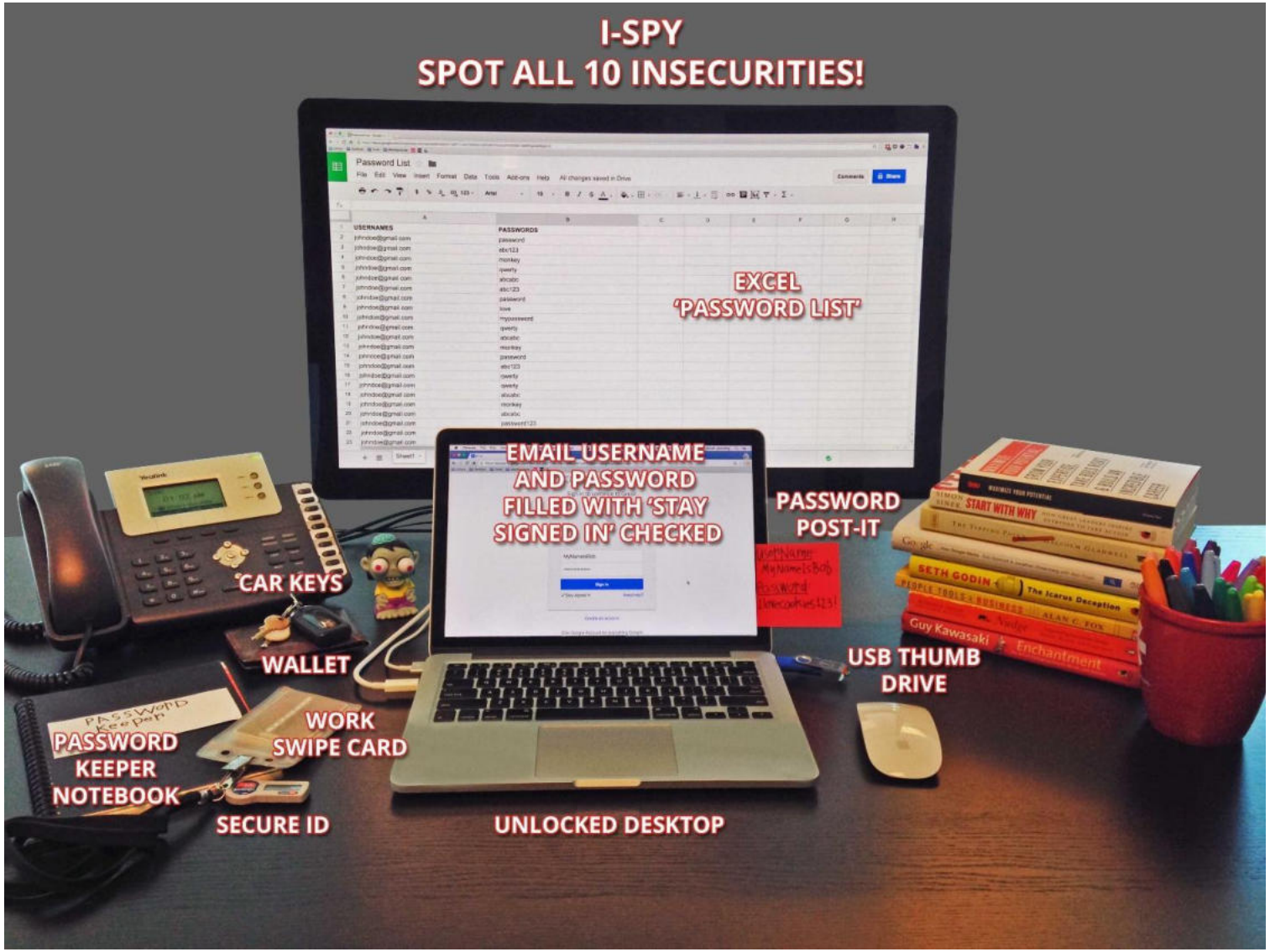
Redundant utilities

- Most facilities need at least electricity, water and telecommunications services
- Ensure that the facility has redundant (i.e. two) sources for the key utilities
- Electricity from two separate substations
- Water from two different main lines
- The redundant utilities enter the building at separate points → no single point of failure
- Underground cabling and pipes
- Separate water pipes from the other utilities → lowered risk of flooding the cabling

Physical and personnel security

INSIDE THE CONTROL ROOM

Workstation security (what not to do!)



Workstations

- Theft protection
- Encrypted file systems
- Biometric sensors and strong passwords → cards might be a better choice compared to passwords
- BIOS password
- Screen orientation → do not allow possible attackers direct view via windows to sensitive screens
- Automatic session locking when the operator leaves the WS
- Keep sensitive documents and plans in locked cabinets

Secure computer workstations

- Locked compartments for
  - Computer cases
  - Screens
  - Keyboard, mouse

- Threats:
  - Locks can be picked
  - Doors can be left unlocked by reckless employees
  - Keys can be lost

https://www.equiptowork.com/products/computer-workstations

Physical port security

- Lock-ins for network cabling
- Port blocking devices for USB ports

https://www.amazon.com/Lindy-RJ45-Port-Blocker-40470/dp/B00F3VBND4

Biometric sensors

- Sensitive workstations need strict access control
- Passwords are not a viable solution
- All types off remote access to workstations should be default deny
- MFA is a possible solution
  - Combined fingerprint and card reader

https://www.acs.com.hk/en/products/131/aet65-smart-card-reader-with-fingerprint-sensor/

THE 5 LEVELS OF PHYSICAL SECURITY

5 Levels of Physical Security

Security level	Security controls (additive)	Examples
Zero	None	high seas; forest; open fields
Minimum	Basic security controls to keep intruders out: cleared space, lighting, fences, walls.	private residences; urban spaces
Low	Security gate, reinforced locks, bars on windows, alarm.	small shops; storage facilities
Medium	Protection against internal and external threats. Internal surveillance monitors employees, customers and guests. High fence, unarmed security guard, loud alarm.	factories; large retail stores; warehouses;
High	24/7 closed-circuit television (CCTV), perimeter alarm system, gates, controlled access in and out, security lighting, armed guards, security dogs, audit, cooperation with law enforcement.	prison; defense; electronics; pharmaceuticals;
Maximum	24/7 security personnel (combat-trained, well-equipped), stricter internal control of movement, tamper-proof alarm.	military installations; embassies; nuclear; gov facilities;

Physical and personnel security

PERSONNEL SECURITY

How to tighten personnel security?

https://gcn.com/articles/2015/03/23/personnel-security?m=1

Control whom you hire…

- Check every prior position
  - Pay special attention to any ‘holes’ in a CV → the candidate might’ve served a prison sentence or worked for a classified organization
- Check key claims in a CV
  - Degrees, certificates
  - Knowledge of languages
  - Criminal record
  - Note: checking creditworthiness is not privacy-proof (!)
- Check all referrers, e.g., past employers
- Check physical and psychical capabilities → control room operators might face highly stressful situations during their work

Secure & monitor your human resources

- Define persons who handle security incidents, either physical or electronic
  - Define chains of notification within departments and/or on floors within your building(s)
- Organize periodic security trainings (in person, not electronic !)
- Frequently send out short notifications about the latest known threats
- Consider offering awarding special prizes to employees who strictly follow the security policy
- Consider additional monitoring specific employees during specifically sensitive periods
  - Decreased salary
  - Reassignment, e.g., to a different department
  - Before known lay-offs

Let them go if they have to leave…

- Control the software and hardware used by the employee
- Do the following synchronously in the physical domain
  - Deny access to any workstations
  - Return company laptop
  - Return keys, magnetic and other cards
  - Escort the employee outside the facility
- Do the following synchronously in the electronic domain
  - Deny and uninstall VPN access
  - Delete and/or deactivate all electronic accounts – some industries do not require an immediate deletion
  - Delete publicly available information about the (past) employee, e.g., delete info from the company's public website
- When a high-risk employee leaves (e.g., OT server room admin) then create backups and increase the sensitivity of security monitoring solutions

Secure flow of human resources

- Well-trained employees
- Vetted contractors
- Proven subcontracted service providers
  - Garbage
  - Raw material logistics
  - Complete products leaving the facilities
- Short trainings for guests

Security onion around the ‘core’

- Ensure that anyone entering the most sensitive parts of the facilities was authenticated multiple times
  - At the entrance/gate
  - At the employee entrance ensuring that there is no piggybacking
    - Floor-to-ceiling turnstile
    - Mantrap with two separate doors and an airlock between them. Both doors require authentication
  - At the inner door to the sensitive part of the facility with multi-factor authentication, surveillance and strict auditing policy

People flow

- Employees
  - Company card with photo and color-coding aligned access level
- Contractors
  - Limited use cards
  - Food delivery services strictly controlled and any changes in schedule or personnel based on prior notification only
- Guests
  - Temporary, one-day cards
  - Receive a basic physical security training on first entry
  - Continuously escorted by at least one employee

Personnel Security Summary

Hiring and onboarding

- Check claims in CV
- Verify via 3rd parties

Employment

- Periodic trainings
- Latest threats bulletins
- Awards
- Monitoring and checkups

Contract termination

- Escort outside facility
- Delete accounts
- Withdraw physical access tokens
- Monitor accessed assets

Conclusion

- Basic definition and data lifecycle
- Attackers and attack types
- Methods
- Physical security
  - Outside the facility
  - Physical access points
  - Inside the facility
  - Inside the control room
- Personnel security in brief

Thank you for your attention!

Eötvös Loránd University (ELTE)
Faculty of Informatics (IK)
Pázmány Péter sétány 1/c
1117 Budapest, Hungary

DATA PROTECTION CHALLENGES AND SOLUTIONS

Lecture #2 in Introduction to Data Security

Imre Lendák, PhD, GICSP

2026
Budapest, Hungary

Presentation outline

- Data inventory and backup
- Cryptography in a nutshell
  - Algorithm types
  - Applied cryptography
- Access control
  - Identity
  - Authentication
  - Authorization
  - Auditing
- Secure software development

DATA INVENTORY AND BACKUP

Data inventory

Example questions asked

- Does the system store personally identifiable information (PII)?
- What is considered data? Is the answer (critical infrastructure) sector-specific?
- Is data management regulated in the CI sector at hand?
- Is information system configuration data?
- Which are the data flows (internal and external)?
- Which artificial intelligence-related artifacts are considered data?

Data inventory

- Physical documents + confidentiality level (public, …, top secret)
- Electronic documents
- Electronic databases (schema + data)
- System configuration (JSON, etc.)
- Binary files (exe, dll, so, etc.)
- Machine learning models + training data + algo implementation + hyper parameters → More on this later (!)

- **Discuss:** What else?

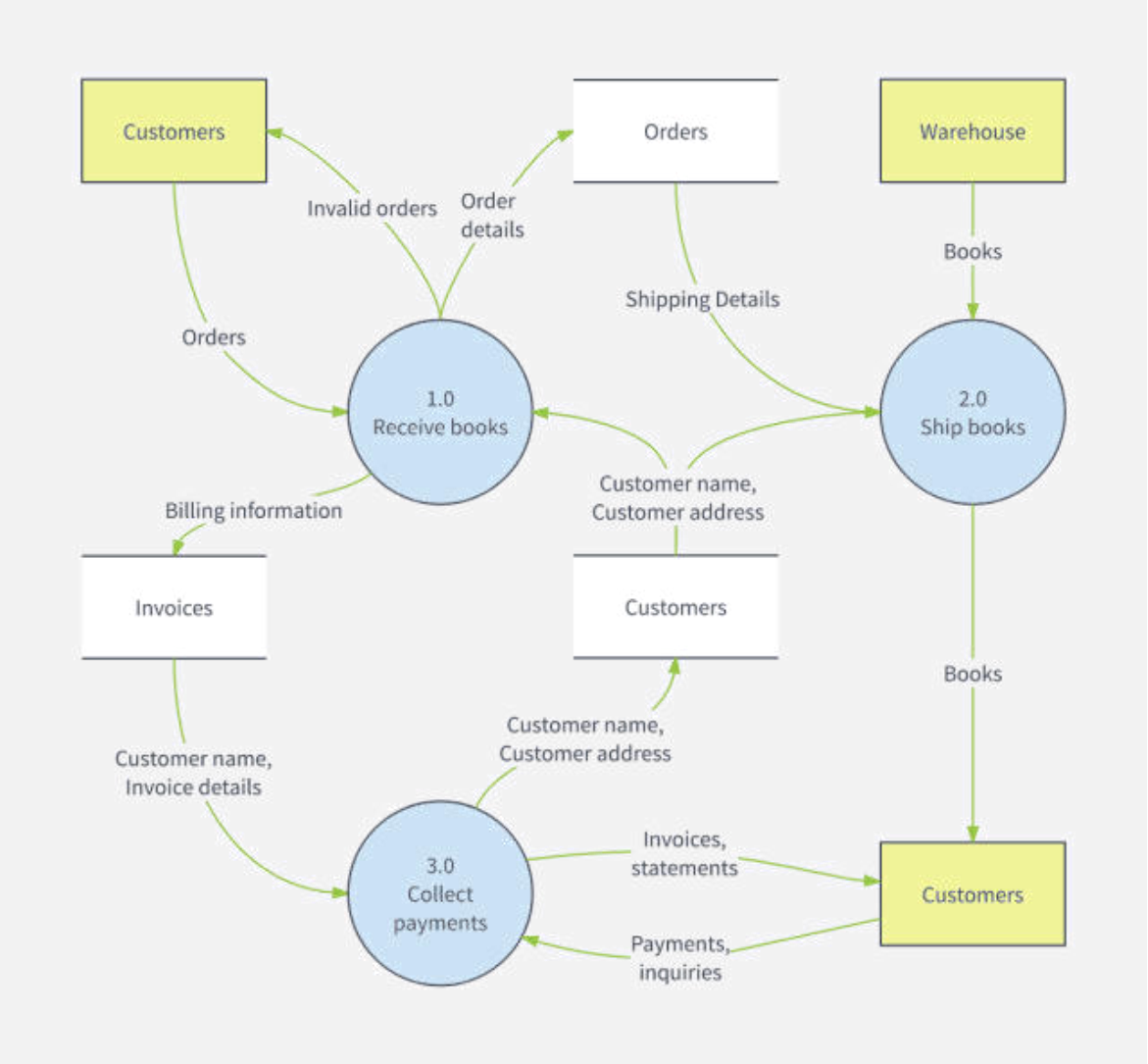
Data inventory template (org specific)

Id	Name	Description	Location	Confidentiality level
E001	Employment contract template		//store/legal/	Internal
P001	Incident Response Plan (IRP)	Physical copy of the IRP	Main archive, cupboard X, drawer Y	Internal
P002	Cybersecurity Strategy	Internal copy	//store/public/it/security	Public
E002	Cybersecurity Strategy	Published version	www.company.com/css	Public

Data at rest: Backup and encryption

- Hot backup solutions:
  - Redundant Array of Independent Disks (RAID)
  - Redundancy levels: in-rack, off-rack, off data center
- Cold backup:
  - Different media (tape, DVD, paper)
  - Offsite if possible
- Password protection e.g., compressed files
- Encrypt files, folders and file systems
  - **Discuss:** Distributed filesystem challenges?

Data in transit: Secure comms architecture



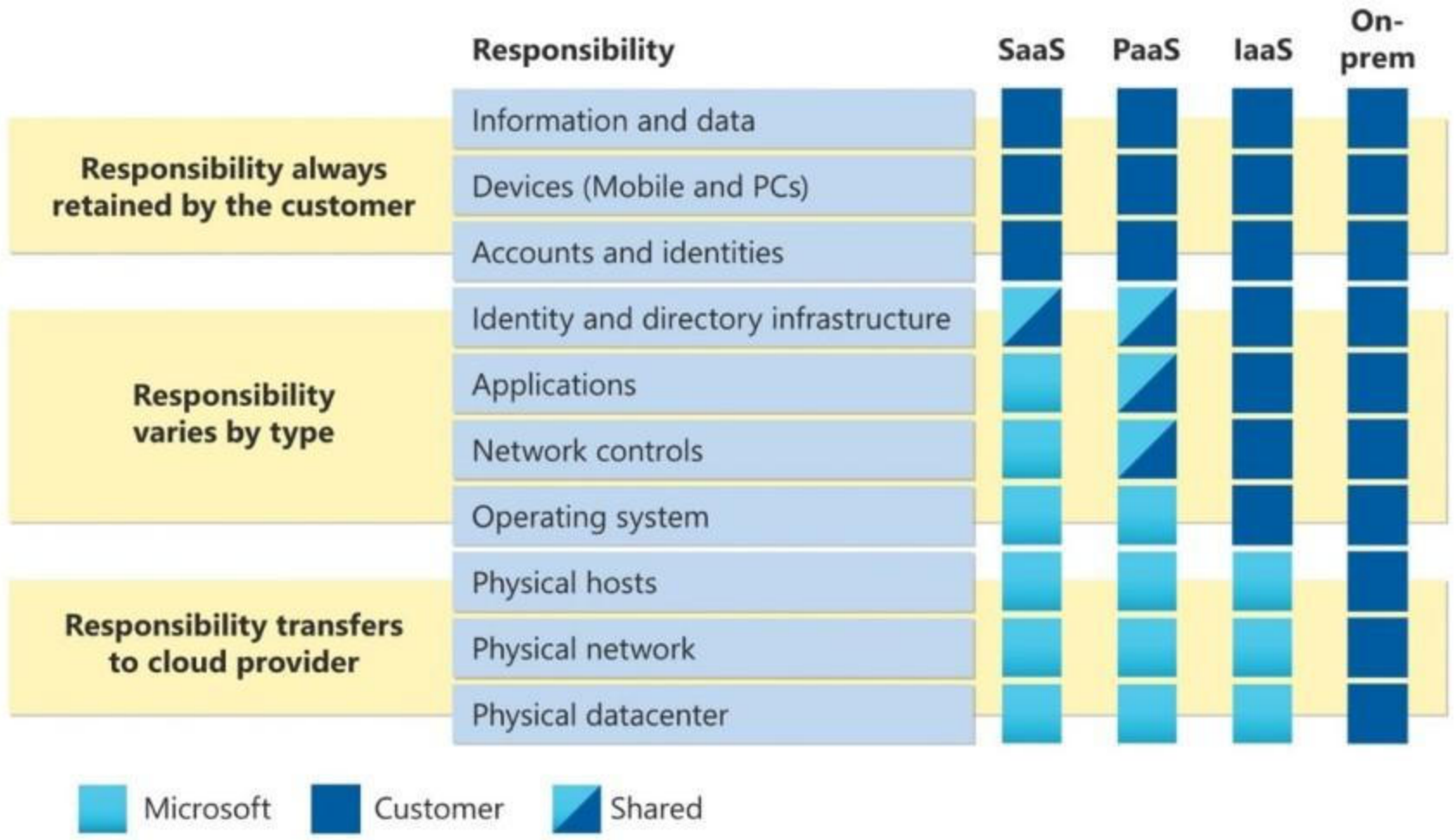
- Identify processes and users
- Identify data stores
- Identify data flows between processes and data stores
- Ensure that the content of each flow is
  - defined,
  - secured in line with security requirements, and
  - audited.

https://remonsinnema.com/2020/09/21/data-flow-diagrams-and-threat-models/

Data in use: Challenges and solutions

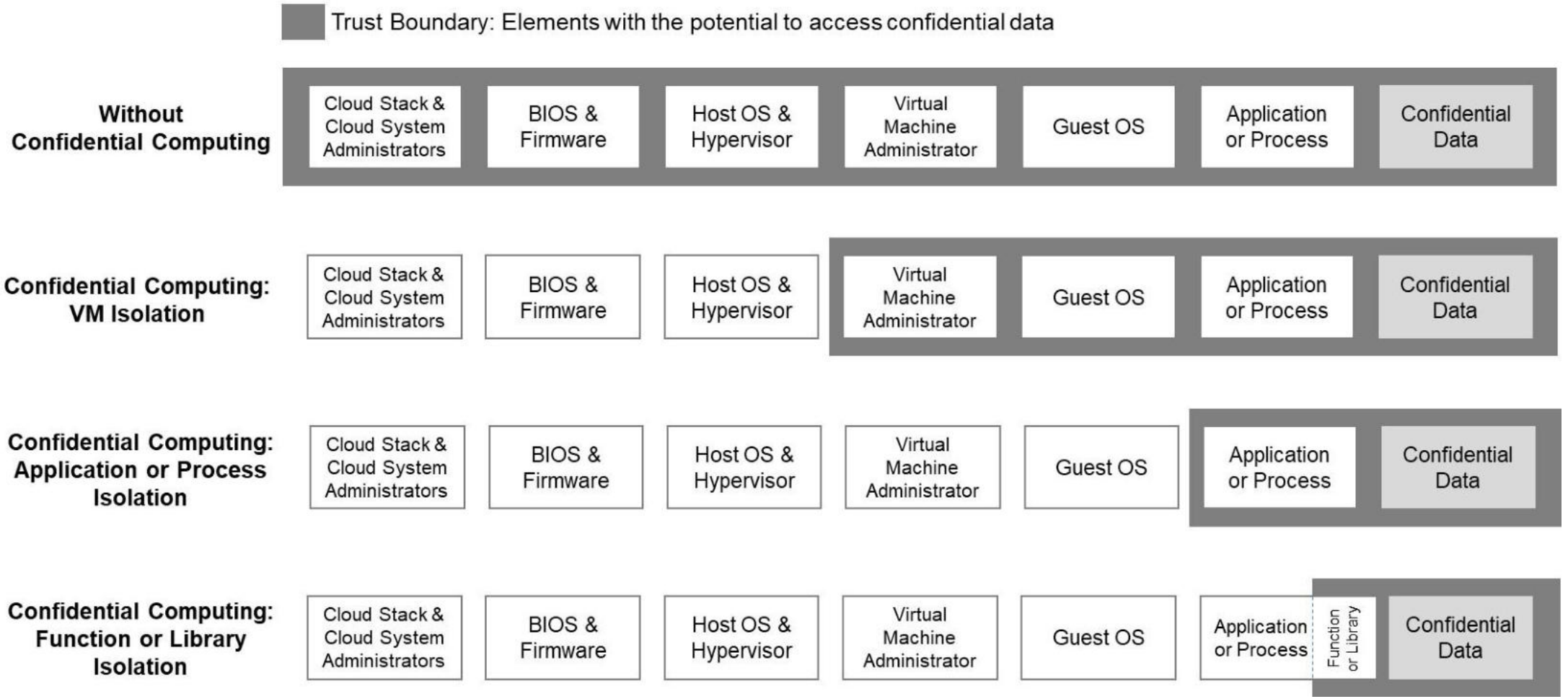
- **DEF:** Data in use includes data accessed, modified and processed by users, applications and devices.
- **Alternate DEF:** Data is in use when it is in a non-persistent digital state, usually the random-access memory (RAM), CPU registers or cache.
- Common security principles and approaches:
  - Choose reliable service providers with ISO 27001 or PCI DSS certificates
  - Zero trust → Trust, but (always) check
  - Strictly monitor data access and keep audit logs
  - Encrypt data wherever possible → Consider the use of homomorphic encryption
- Modern CPU architectures implement components of cryptographic algorithms
- **Trusted execution environment (TEE)** technologies prevent unauthorized access or modification of applications and data while in use
  - Example tech: AMD Secure Encrypted Virtualization - Secure Nested Paging (AMD SEV-SNP), Intel Software Guard Extensions (Intel SGX) & Trust Domain Extensions (Intel TDX)

Data in use: Cloud-based challenges



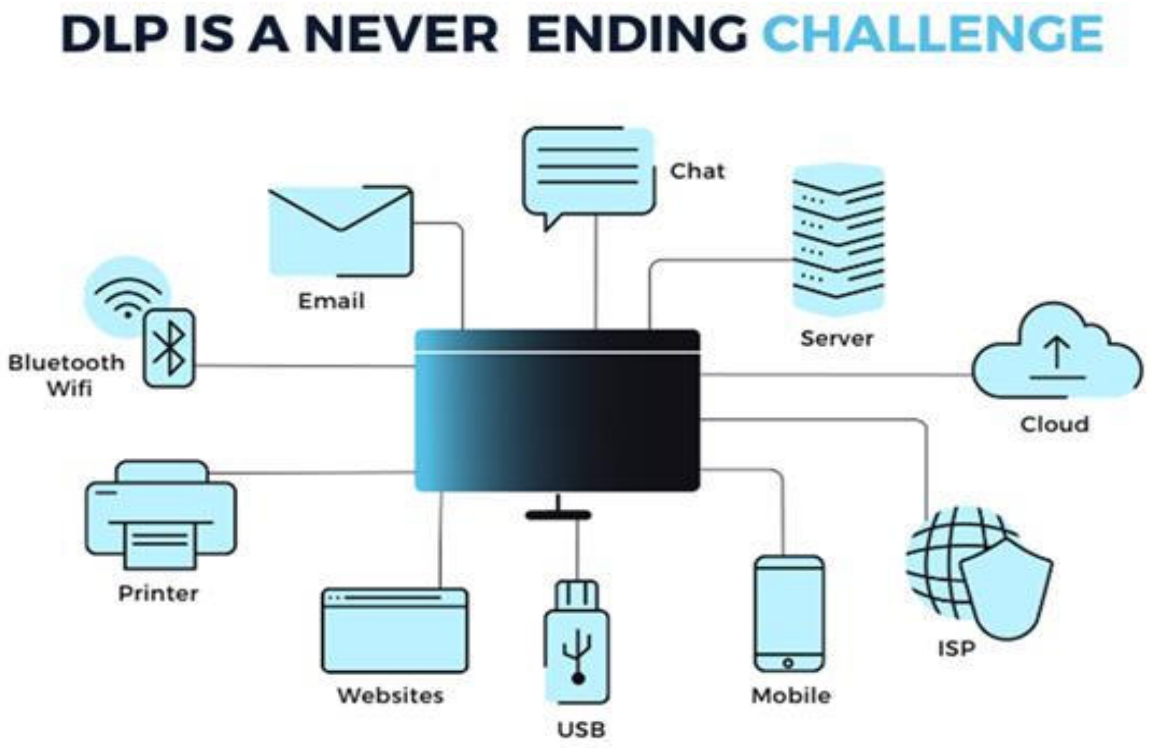
https://www.microsoft.com/en-us/security/blog/2023/07/05/11-best-practices-for-securing-data-in-cloud-services/

Data in use: Confidential computing & TEEs



https://en.wikipedia.org/wiki/Confidential_computing

Data loss prevention (DLP)



https://luxsci.com/blog/futureproof-your-data-loss-prevention-strategy.html

- **DEF:** Data loss prevention (DLP) solutions allow system operators to **detect and respond to data breaches**.
- DLP functionality:
  - Enforce least privilege and zero trust → Trigger additional auth challenges.
  - Warn users when violating policy.
  - Terminate unauthorized access.
  - Flag suspicious behavior for the security team.
- Types of DLP based on location:
  - Endpoint = monitor data on endpoints (laptop, server, mobile)
  - Network = analyze data exchanged in computer networks
  - Cloud = monitor data accessed by cloud services and/or stored in the cloud

CRYPTOGRAPHIC ALGORITHM TYPES

List of sub-topics

Algorithm types

- Symmetric
- Asymmetric
- Hash functions

- Homomorphic encryption
- Post-quantum cryptography

Systems and applications

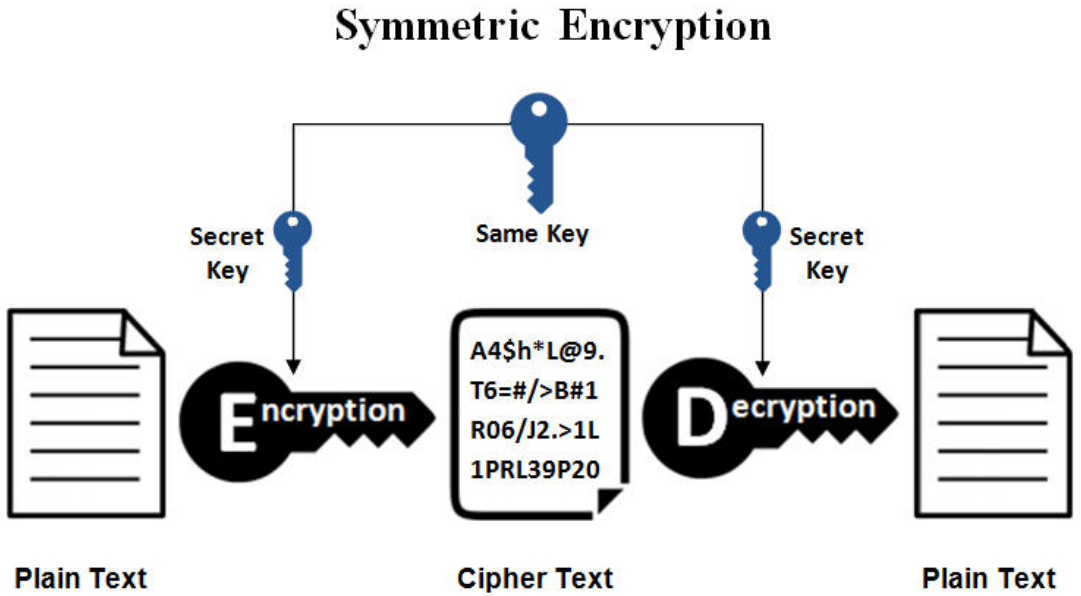
- Public Key Infrastructure (PKI)
- Perfect forward secrecy

- Data in transit
  - Digital certificates
  - HTTPS
  - Digital signatures
- Data at rest
  - File encryption
- Data in use

Symmetric cryptography

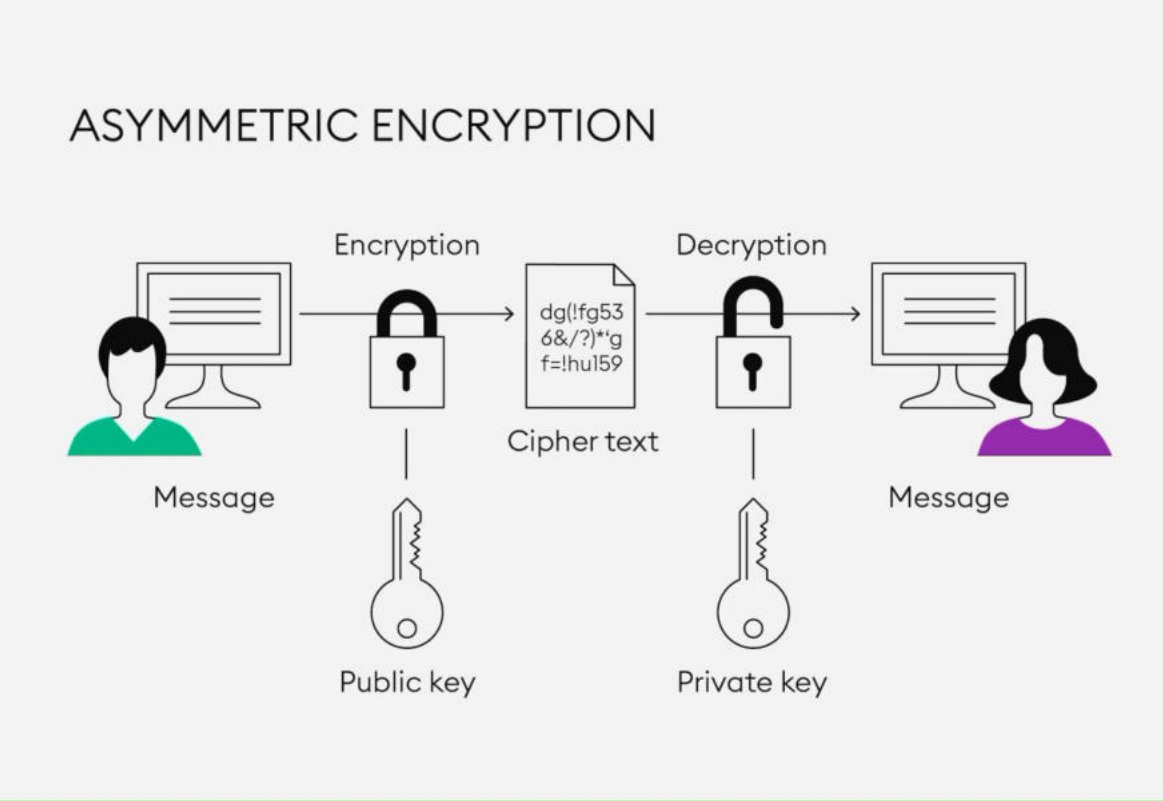
- **DEF:** In symmetric cryptography both (or more) parties use the same key to encrypt and decrypt
- Mathematical foundation:
  - Sequence of inexpensive transformations with use of cryptographic key

- **Example algo(s):** AES
- **Key challenges:** key exchange, forward secrecy
- **Usage:** WiFi, HTTPS



https://www.ssl2buy.com/wiki/symmetric-vs-asymmetric-encryption-what-are-differences

Asymmetric cryptography

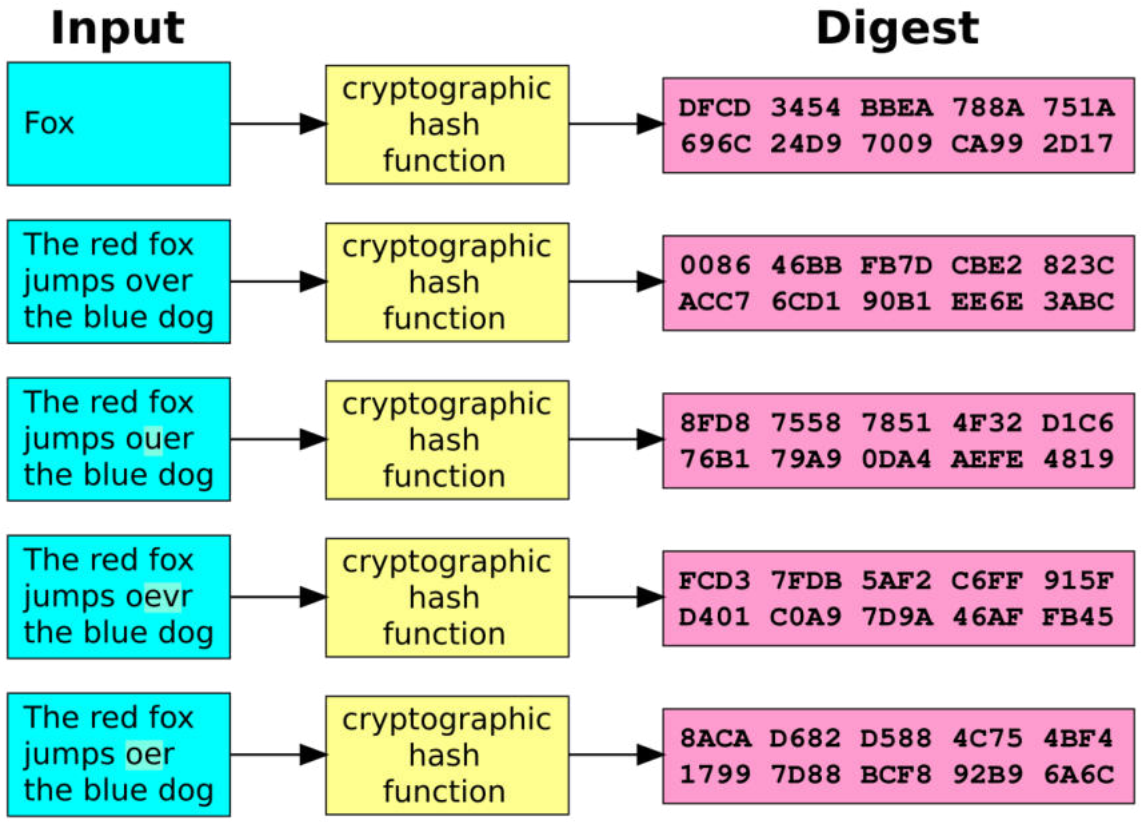


https://www.bitpanda.com/academy/en/lessons/what-is-asymmetric-encryption/

- **DEF:** In asymmetric cryptography a mathematically bound pair of keys is used for data encryption
- Mathematical foundation:
  - Trapdoor functions → Inexpensive to calculate in one way, but inverse function is difficult to compute
  - Example: Prime number factoring
- **Example algo(s):** RSA
- **Key challenges:** quantum computers, computation cost
- **Usage:** HTTPS, digital signatures

Cryptographic hash functions

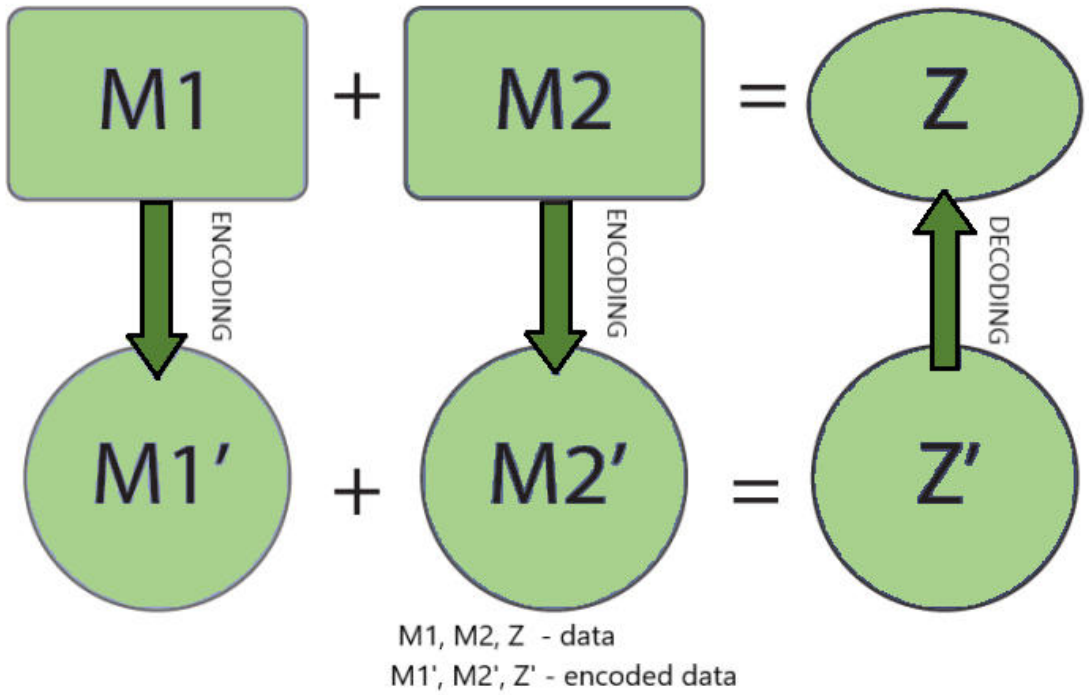
- **DEF:** Cryptographic hash functions transform input data of arbitrary length to fixed-length digest (aka 'hash')
  - Examples: MD5 (128 bits), SHA2 (up to 512 bits)
  - Usually serialized to strings of hexadecimal numbers
- Key features:
  - Small change in input → significant change in output
  - Pre-image resistance
  - 2nd pre-image resistance & collision resistance
- **Discuss:** Challenges related to fixed-length output?
- Application domains:
  - Storing passwords
    - Salt = String concatenated to the input
  - Data structures for fast lookup
  - Digital signatures
  - **Discuss:** Any other application domains? Which?



https://en.wikipedia.org/wiki/Cryptographic_hash_function

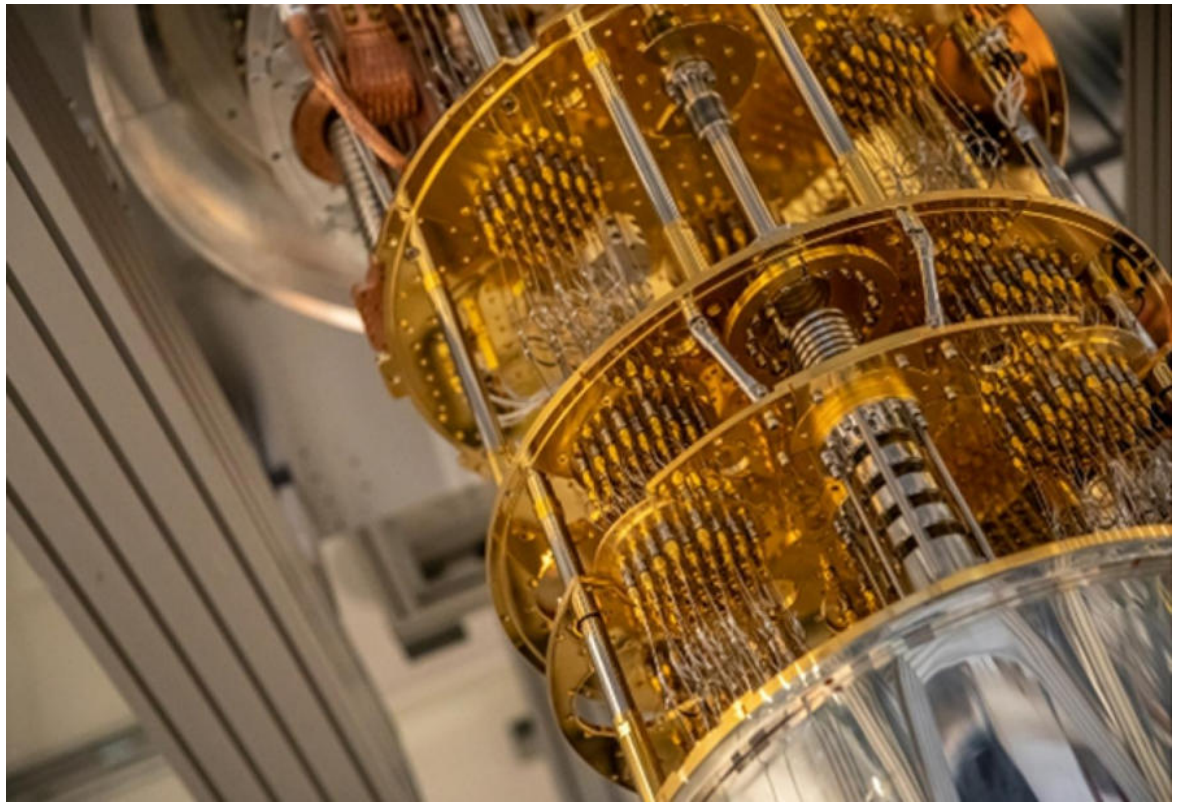
Homomorphic encryption

- Homomorphism comes from the ancient Greek language:
  - homos (ὁμός) meaning "same" and
  - morphe (μορφή) meaning "form" or "shape“
- Allows operations to be performed over encrypted data.
- Data intermediators will not have access to encrypted data.
- 1978: Ronald Rivest, Leonard Adleman and Michael Dertouzos suggested the concept of homomorphic encryption for the first time.
- 2009: Craig Gentry from IBM has presented the first encryption system "fully homomorphic“.
- **Examples:** RSA



Post-quantum cryptography

- Large-scale quantum computers will ‘break’ many traditional cryptosystems (RSA, DH, DSA, ECDSA, ECC, HECC, etc.)
- Post-quantum cryptographic algorithms use novel mathematical foundations:
  - Lattice-based
  - Code-based
  - Multivariate-based
- 2015-today: NIST post-quantum standardization process & competition

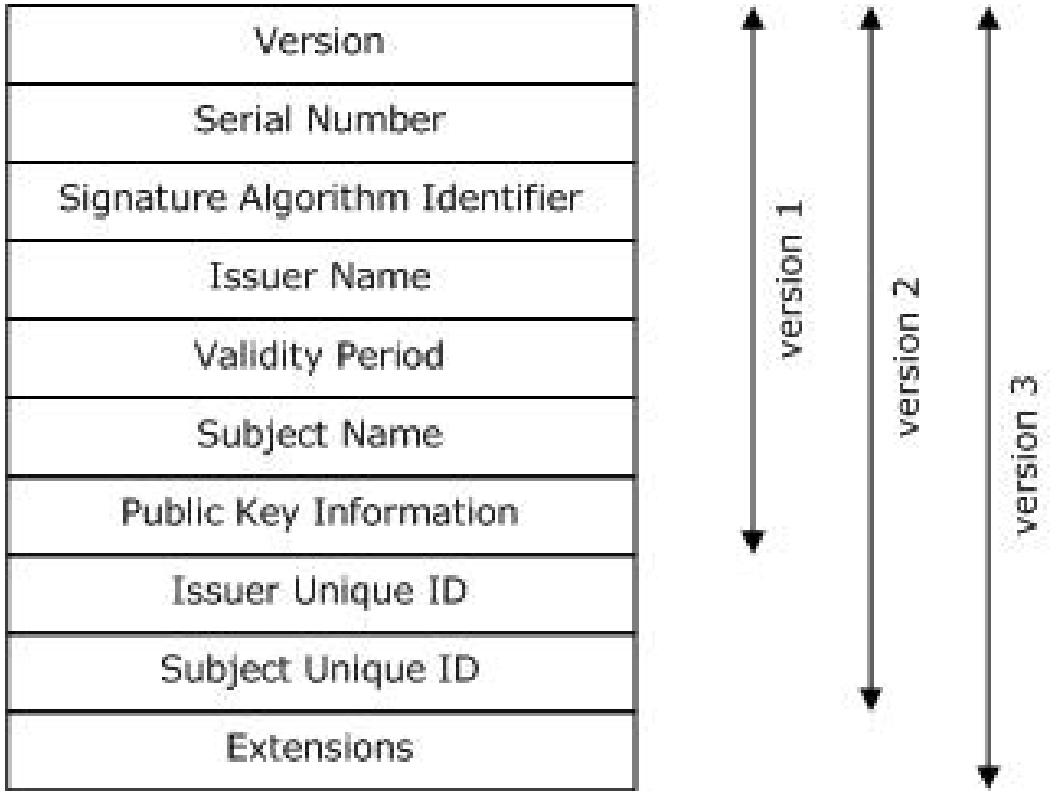


https://www.energy.gov/science/doe-explainsquantum-computing

APPLIED CRYPTOGRAPHY

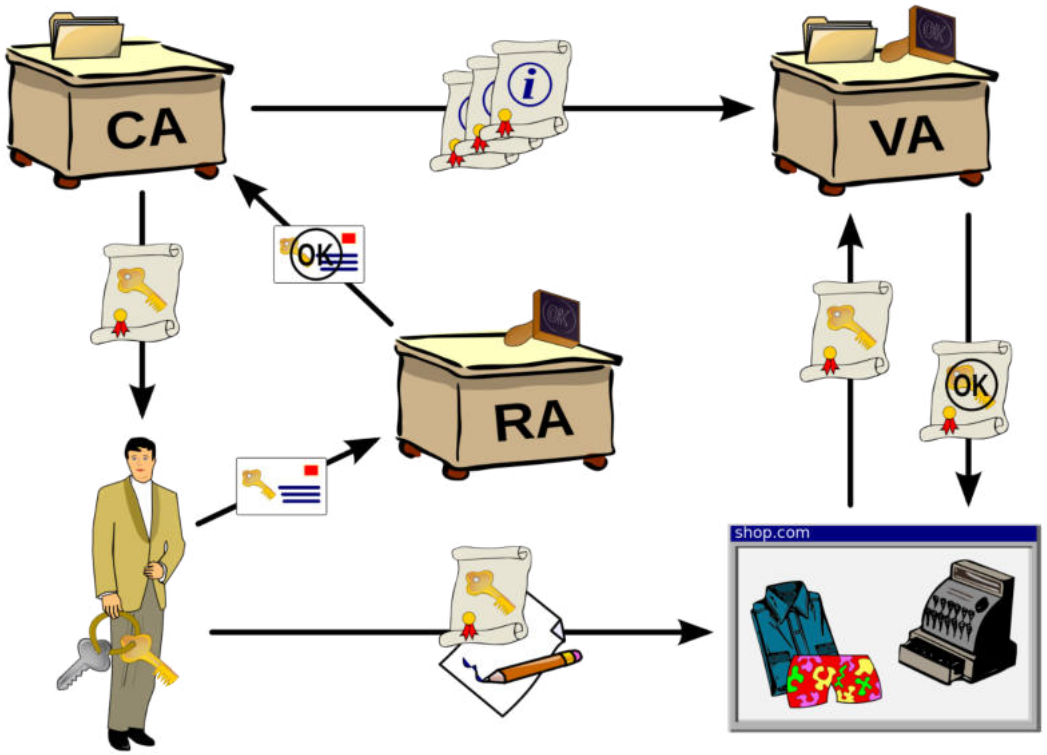
Digital certificates

- **DEF:** A digital certificate is a third-party validated proof of user, device, server, or website authenticity.
- Minimum contents:
  - Distinguished name of certificate owner
  - Public key of the owner
  - Digital signature of issuing 3rd party
  - Validity period (from-to dates)
- Example: X.509 (ITU standard)



https://learn.microsoft.com/en-us/windows/win32/seccertenroll/about-x-509-public-key-certificates

Public Key Infrastructure (PKI)



- **DEF:** A Public Key Infrastructure (PKI) creates, manages, distributes, uses, stores, and revokes **digital certificates** and public-keys
- PKI actors (apart from customer and service provider)
  - Certificate Authority (CA): Issues certificates
  - Registration Authority (RA): Registers entities
  - Validation Authority (VA): Validates entities
- **NOTE:** PKI is key to securing data transferred over public networks

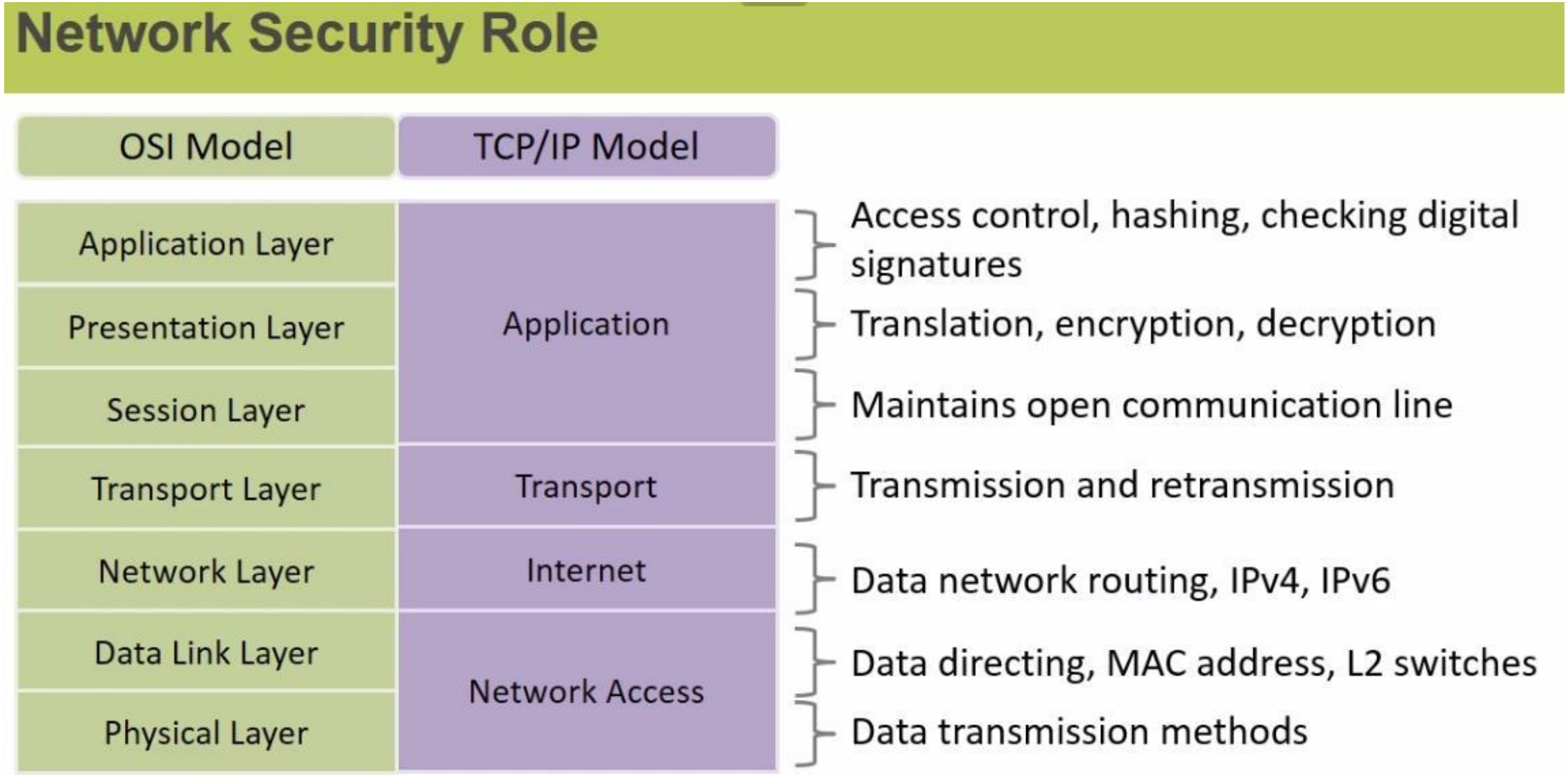
Data in transit: Perfect forward secrecy

- **Discuss challenge:**
  1. Attacker eavesdrops on encrypted data today,
  2. stores the data,
  3. and decodes later when the encryption key is found with key leakage, stronger hardware or cryptoanalysis solutions become available.

- Perfect forward secrecy (PFS) enforces a new key exchange process in line with system configuration (usually for every communication session)
- **Discuss:** Differences between PFS and FS?

https://sectigostore.com/blog/mitigating-session-data-exposure-perfect-forward-secrecy-explained/

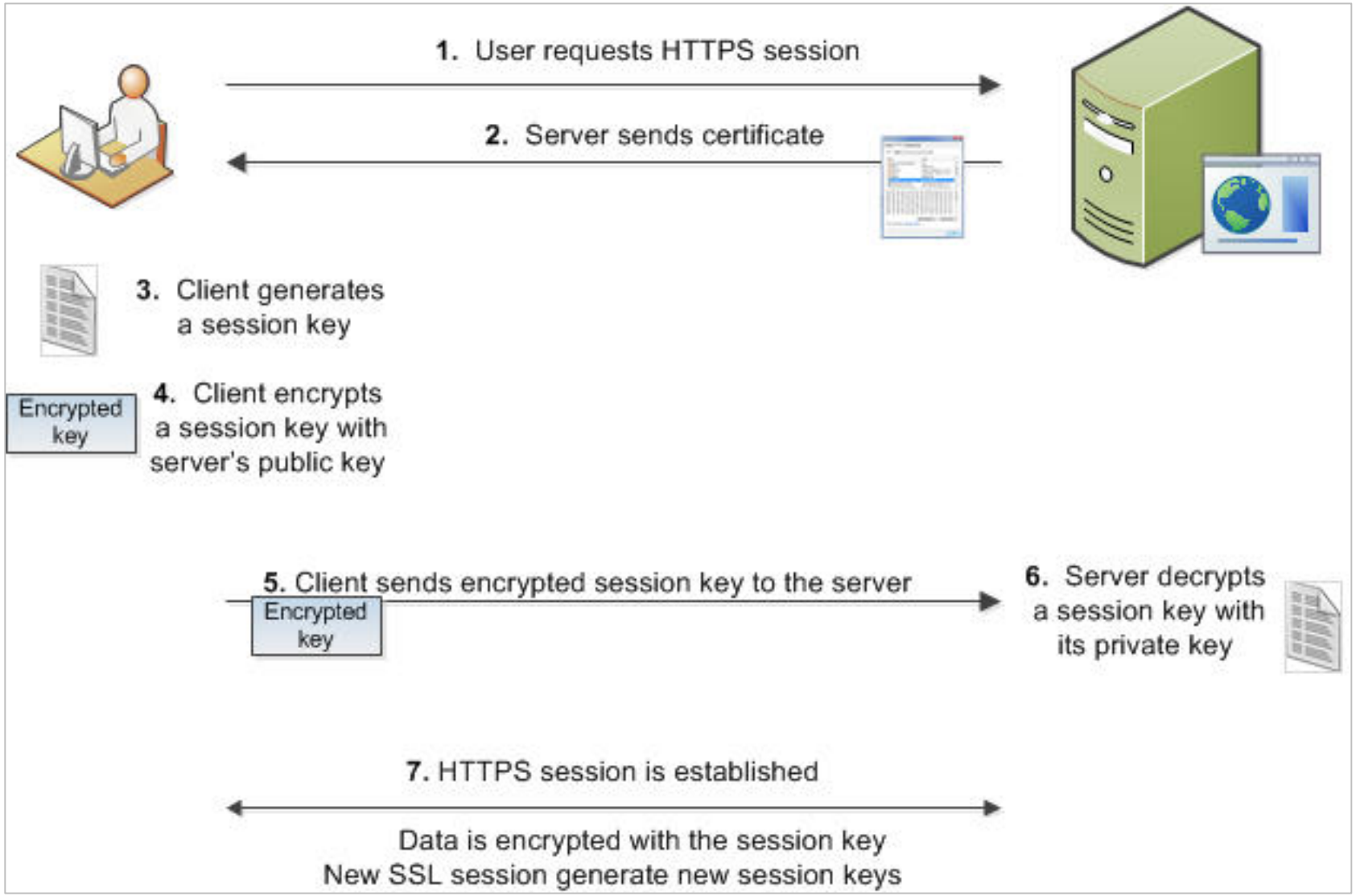
Data in transit: Crypto in the TCP/IP stack – 1



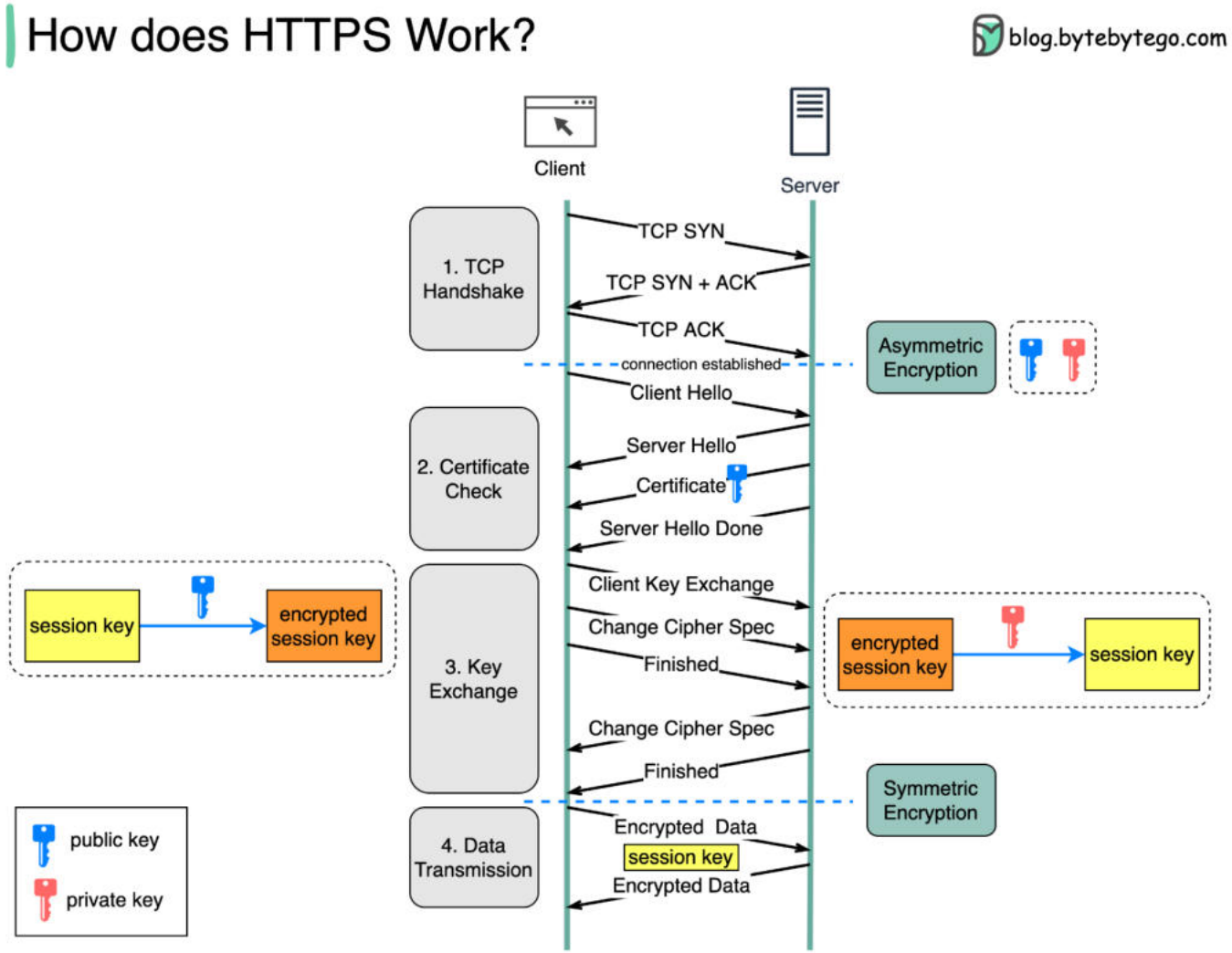
Data in transit: Crypto in the TCP/IP stack – 1

Application	• Secure Shell (SSH) • HTTPS (extension of HTTP)
Transport	• Transport Layer Security (TLS) • Secure Socket Layer (SSL)
Internet	• IPsec (adds authentication and encryption)
Network Access	• 802.1X (port-based network access control)

Data in transit: HTTPS (1)



Data in transit: HTTPS (2)

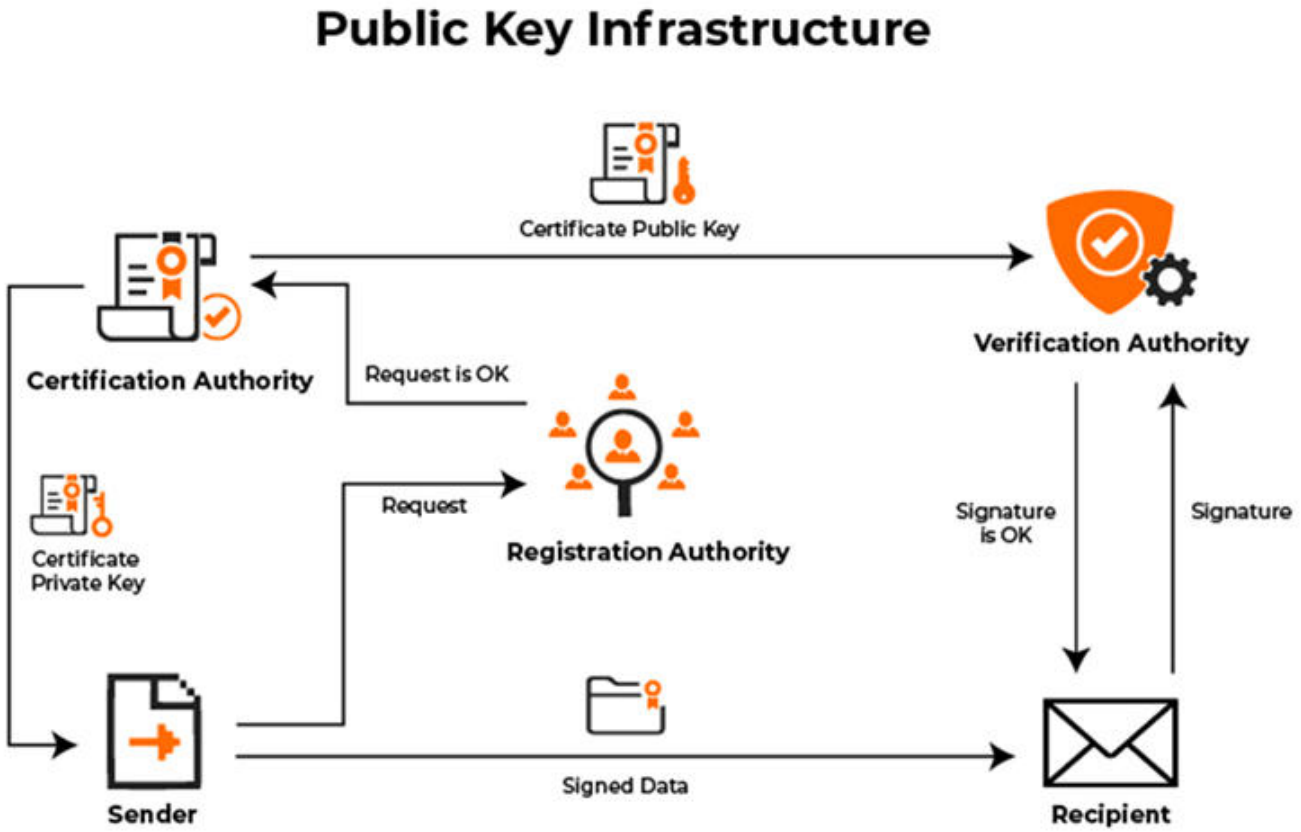


Data in transit: Digital signatures

Step-by-step view

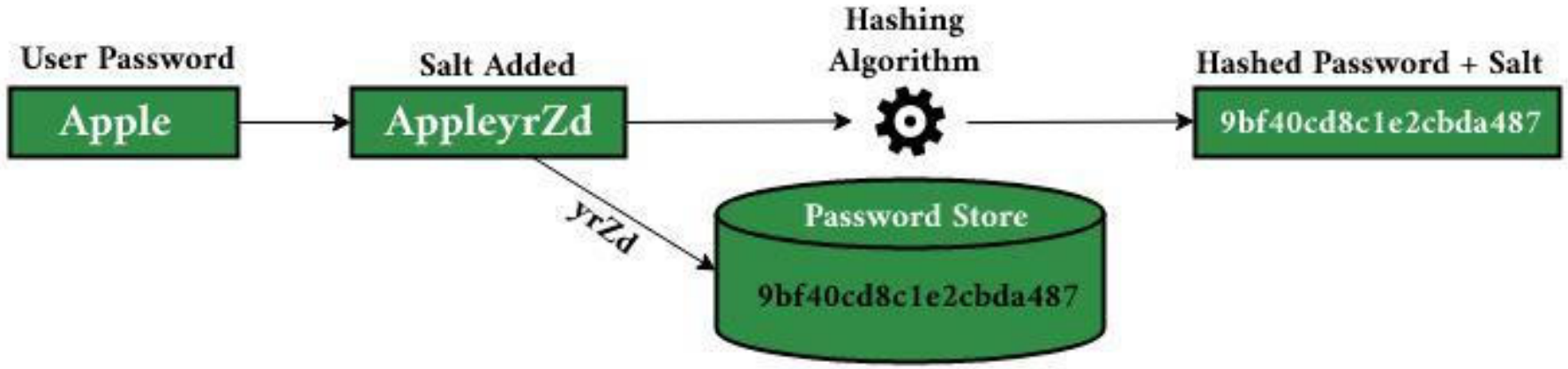
- Apply digital signature
  - Calculate message hash
  - Encrypt message hash with private key
- Send message and signature via the open Internet
- Check digital signature
  - Calculate message hash
  - Decrypt received message hash
  - Compare hash values

- **Discuss:** Which guarantees are provided?



Data at rest: Password storage & salt

- Rainbow table = Large database of known (password, hash) pairs
- Password salt = Concatenate random string of characters to plaintext password before hashing




```
sai:$6$YTJ7JKnfsB4esnbS$5XvmYk2.GXVWhDo2TYGN2hCitD/wU9Kov.uZD8xsnleuf1r0ARX3qodIKiDsdoQA444b8IMPMOnUWDmVJVkeg1:19446:0:99999:7:::
```

https://www.geeksforgeeks.org/what-is-salted-password-hashing/
https://www.cyberciti.biz/faq/understanding-etcshadow-file/

Data at rest: Encryption

- Encryption solutions:
  - Levels: file, folder, file system
  - **Discuss:** OS type vs file encryption?
  - **Discuss:** Distributed filesystem encryption? Is it done? How?

ACCESS CONTROL (AAAA)

Identity

- **DEF:** Identification allows information systems to differentiate users, documents, processes and data.
  - Tasks: generate, assign and check identities
- Identities can be tied to entities in the physical or in cyber space:
  - In the physical space: persons with different roles (employee, guest, subcontractor), paper-based documents, identifiable shipments of goods (or data!).
  - In cyberspace: electronic identities of users, documents or services
- Identification does not rely on strong proofs which would validate who is who – and can be based on (relatively) weak (and easy to forge) claims as follows:
  - Physical space: driver's license
  - Cyberspace: unique identifier presented
- Truly unique IDs can be tricky to generate:
  - Discuss: IP & MAC addresses, UUIDs, human-readable?

Authentication

- **DEF:** Authentication requires entities in information systems to provide proof of their claimed identities.

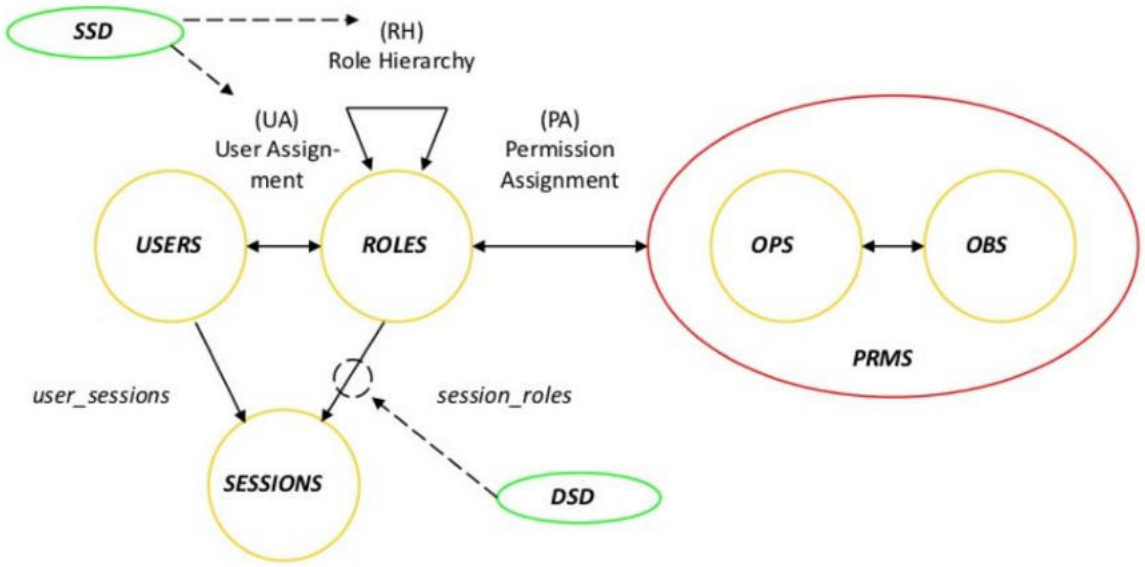
- Multifactor authentication (MFA) further strengthens authentication measures by requiring two different means of identity verification (according to NIST):
  - (i) something you know → Passwords: strong, one-time
  - (ii) something you have → Swipe cards: issued in medium and higher physical security environments
  - (iii) something you are → Biometric sensors: fingerprint sensor, typing pattern, face recognition

- **Discuss:** How to do MFA for electronic-only identities e.g., processes run by an OS on a specific virtual machine or in a container?

Authorization

- **DEF:** Authorization is the process in which entities in an information system (IS) are assigned and controlled access to IS assets
  - Authorization is key in both data confidentiality and integrity (CI of the CIA triad)

- Access control models:
  - Mandatory
  - Discretionary
  - Role-based → Depicted on the right (!)
  - Attribute-based



Entities in role-based access control (RBAC)

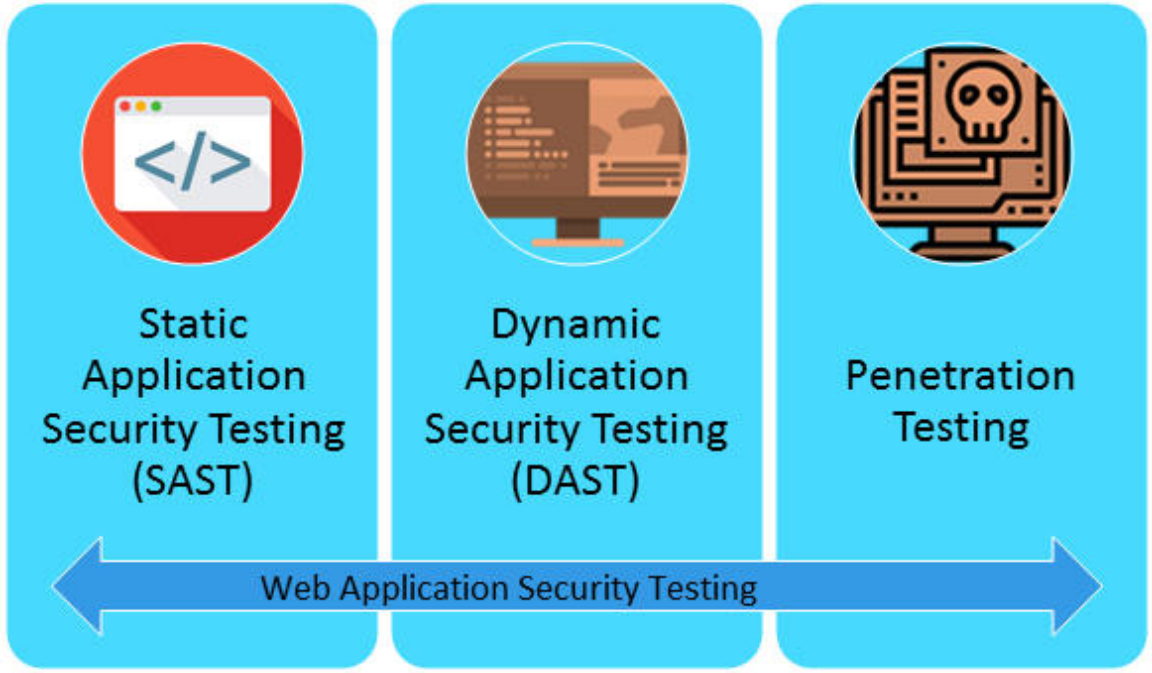
Auditing & Accounting

- An **audit log** is a chronological list of relevant activities in an information system (IS).
- Value of audit logs:
  - Regulatory compliance
  - Optimize IS operations
  - Detect anomalies → Detect malicious insider activity
  - Detect trends and changes
- **Accounting** keeps track of activities performed by authenticated entities.
- Value of accounting:
  - Allows service providers to bill services provided
  - Allows provider and consumer to check service quality → Service Level Agreement (SLA)

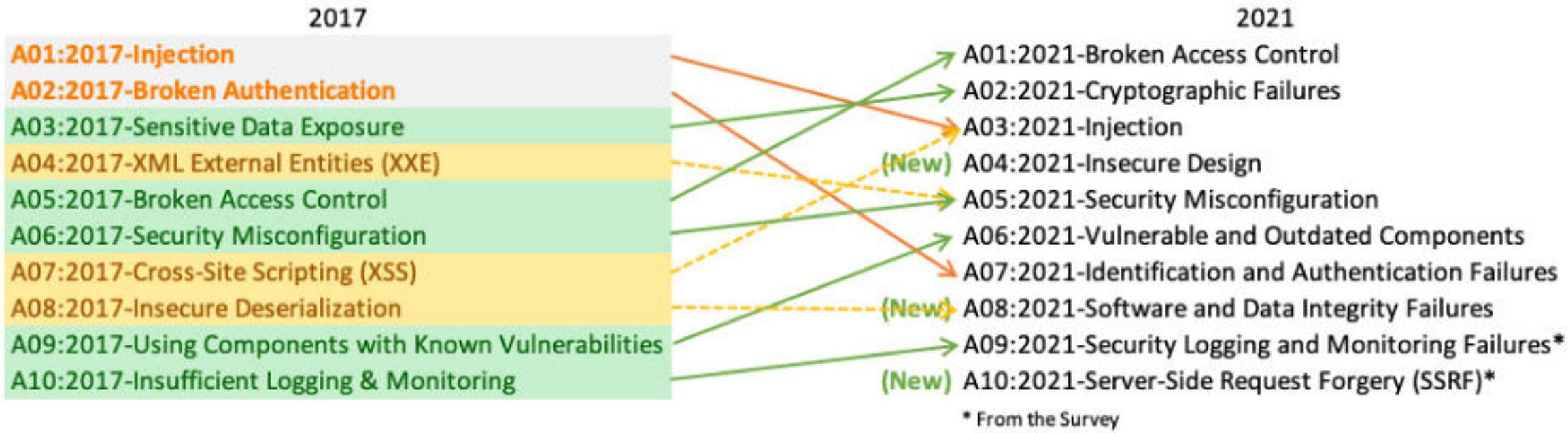
SECURE SOFTWARE DEVELOPMENT

Secure software development intro

- Secure Development Lifecycle (SDLC)
  - Security incorporated in each lifecycle stage
- Application Security Testing (AST) levels
  - Static AST: Input = source code
  - Dynamic AST: Input = binary code, configuration files, test case definitions
  - Interactive AST: Input = modified binary with probes + dynamic
  - + Penetration Testing



OWASP Top 10 Web Application Security Risks



https://owasp.org/www-project-top-ten/

OWASP Top 10 API Security Risks – 2023

API1:2023 - Broken Object Level Authorization	• APIs tend to expose endpoints that handle object identifiers, creating a wide attack surface of Object Level Access Control issues. Object level authorization checks should be considered in every function that accesses a data source using an ID from the user.
API2:2023 - Broken Authentication	• Authentication mechanisms are often implemented incorrectly, allowing attackers to compromise authentication tokens or to exploit implementation flaws to assume other user's identities temporarily or permanently. Compromising a system's ability to identify the client/user, compromises API security overall.
API3:2023 - Broken Object Property Level Authorization	• This category combines API3:2019 Excessive Data Exposure and API6:2019 - Mass Assignment, focusing on the root cause: the lack of or improper authorization validation at the object property level. This leads to information exposure or manipulation by unauthorized parties.
API4:2023 - Unrestricted Resource Consumption	• Satisfying API requests requires resources such as network bandwidth, CPU, memory, and storage. Other resources such as emails/SMS/phone calls or biometrics validation are made available by service providers via API integrations, and paid for per request. Successful attacks can lead to Denial of Service or an increase of operational costs.
API5:2023 - Broken Function Level Authorization	• Complex access control policies with different hierarchies, groups, and roles, and an unclear separation between administrative and regular functions, tend to lead to authorization flaws. By exploiting these issues, attackers can gain access to other users' resources and/or administrative functions.
API6:2023 - Unrestricted Access to Sensitive Business Flows	• APIs vulnerable to this risk expose a business flow - such as buying a ticket, or posting a comment - without compensating for how the functionality could harm the business if used excessively in an automated manner. This doesn't necessarily come from implementation bugs.
API7:2023 – Server-Side Request Forgery	• Server-Side Request Forgery (SSRF) flaws can occur when an API is fetching a remote resource without validating the user-supplied URI. This enables an attacker to coerce the application to send a crafted request to an unexpected destination, even when protected by a firewall or a VPN.
API8:2023 - Security Misconfiguration	• APIs and the systems supporting them typically contain complex configurations, meant to make the APIs more customizable. Software and DevOps engineers can miss these configurations, or don't follow security best practices when it comes to configuration, opening the door for different types of attacks.
API9:2023 - Improper Inventory Management	• APIs tend to expose more endpoints than traditional web applications, making proper and updated documentation highly important. A proper inventory of hosts and deployed API versions also are important to mitigate issues such as deprecated API versions and exposed debug endpoints.
API10:2023 - Unsafe Consumption of APIs	• Developers tend to trust data received from third-party APIs more than user input, and so tend to adopt weaker security standards. In order to compromise APIs, attackers go after integrated third-party services instead of trying to compromise the target API directly.

https://owasp.org/API-Security/editions/2023/en/0x11-t10/

Conclusion

- Data inventory and backup
  - Data at rest, in transit and in use
  - Data loss prevention (DLP)
- Cryptography intro
  - Algorithm types
  - Cryptography in action
- Access control
  - Identity
  - Authentication
  - Authorization
  - Auditing
- Secure software development teaser

Additional references

- Dan Boneh, “Cryptography I”, Coursera course, https://www.coursera.org/learn/crypto
- NIST Computer Security Resource Center, “Glossary”, https://csrc.nist.gov/glossary
- Wikipedia, “Confidential computing”, https://en.wikipedia.org/wiki/Confidential_computing
- OWASP, “Top 10 Web Application Security Risks”, https://owasp.org/www-project-top-ten/

Thank you for your attention!

Eötvös Loránd University (ELTE)
Faculty of Informatics (IK)
Pázmány Péter sétány 1/c
1117 Budapest, Hungary

DATA SECURITY ACROSS CI SECTORS

Lecture #3 in Introduction to Data Security

Imre Lendák, PhD, GICSP

2025
Budapest, Hungary

Presentation outline

- Privacy in a nutshell
- Critical infrastructure sectors
  - USA and the CISA
  - Europe and ENISA
- Legal framework and standards
  - The path to NIS2
- Risk management
- Career paths in and around data security
  - European Cybersecurity Skills Framework (ECSF) by ENISA

PRIVACY IN A NUTSHELL

Aspects of data privacy (simplified)

- **(Sensitive) Personally identifiable data (PII)**
  - The objects which need protection
- **Data owners or subjects**
  - Subjects own the objects (i.e., the data)
  - Types of data owners: person, company, government body
- **Data controller** – collects data with certain goal
- **Data processor** – organization or person analyzing the data
- **Controlled access**
  - Determines who can access the data
  - Delegation of data controller or processor rights complicate matters
- **Authentication** of persons, (electronic) identities and attributes

The beginnings of data privacy

- 1974 – Willis Ware (RAND Corporation) report on regulating data security in federal institutions in the USA:
  - **Limited acquisition** of minimum data necessary to achieve stated goals.
  - **Limited use** of data only in the originally designed manner planned prior to acquisition.
  - **Data security** = Adequate security controls to protect the data collected and processed.
  - **Transparency** of data collection and processing towards data owners (subjects).
  - **Subject involvement** = Data owners are allowed to access, modify or delete their data.
  - **Data quality** = Data owners are allowed to modify low quality data.
  - **Legal liability** = The representatives and employees of data collectors and processors have legal liability in relation to the data.
- The Privacy Act was enacted in 1974 based on the Ware report
  - Regulated data privacy of data handled by federal bodies

Data privacy in the 2020s (own grouping)

- Data lifecycle:
  - **Data collection** with consent (!)
  - **Minimal data**
  - **Retention policy** – limited durability
  - **Archival** of outdated data
  - **Deletion** of old data (or court order)
- Subject involvement:
  - **The right to modify data**
  - **Data quality**
- Data use
  - **Limited usage** only for stated goals
  - **Transparency** towards subjects (maybe collectors and processors as well)
  - **Legal liability** of representatives and employees
- Parties accessing the data
  - **Access control** limited to need to know
  - **Trainings** for employees at data collectors and processors
  - **Disclosure** strictly limited and based on authorization
- Data security measures
  - **Encryption**
  - **Anonymization** via removing or changing PII
  - **Audit data access**
  - **Copy management** – backups are ok, additional copies strictly limited

Data privacy in the EU leading to the GDPR

- 1981: Council of Europe Convention No. 108 on data protection
- 1995: EU 96/46/EC data privacy policy
  - Legal liability, fair procedures and transparency
  - Use for specific, predefined goals only
  - Data minimization
  - Precision
  - Limited storage

EU GDPR

- DEF: Regulation (EU) 2016/679 (General Data Protection Regulation – GDPR) on the protection of natural persons with regard to the processing of personal data and on the free movement of such data.
  - In brief: Obliges data controllers and processors located anywhere in the World to protect private data collected about any citizen of the EU
  - Significant penalties ranging up to 4% the total worldwide annual turnover in the preceding financial year or 20 Million EUR (whichever is higher)
- Adopted by the European Parliament and of the Council: April 27, 2016
- Entry into force: May 25, 2018

CRITICAL INFRASTRUCTURE SECTORS

Definitions

- **DEF:** An **infrastructure** is the set of fundamental facilities and systems that support the sustainable functionality of households and firms
- Infrastructure **characteristics**
  - They usually consist of distributed networks of loosely coupled systems
  - They are man-made
  - They are used in the generation, transport and distribution of services or goods
- **DEF:** **Critical infrastructures (CI)** are infrastructures whose incapacitation or destruction would have a debilitating effect on (national/regional) security, national economic security, national public health or safety, or any combination thereof
  - The CI concept was formalized after the **Marsh report (1997)** partially triggered by the Tokyo subway attacks and especially after the 9/11 terrorist attacks
  - In earlier publications **critical infrastructures** and **key assets** were identified separately – in the latest publications they are (usually) consolidated
  - Discussion point: Can you name local or regional key assets?
- **Discuss:** Why is the infrastructure sectors relevant in the (data) security context?

CI characteristics

- **Common CI characteristics**
  - Large-scale production/generation facilities, e.g., dams, large-scale industry, wastewater management plants
  - Long-distance transmission facilities, e.g., long-haul air, highways, electric power transmission
  - Distribution facilities, e.g., for electricity, water, postal services, telecommunications (mobile, landline, internet access)
  - CI can often be modeled with mathematical graphs
- Key **CI differences**
  - Large-scale outages in lifeline CI are immediately felt and have detrimental effects on society
  - Some CI are not directly interfacing with end customers, e.g., dams, nuclear facilities, various critical production subsectors
  - Different monetization of services, e.g., monthly bills for electricity, internet and water vs food, transport
- CI are interconnected into a **system-of-systems**
  - There are significant dependencies between CI, e.g., the regular functioning of internet services, finances and even healthcare are hard to imagine without electricity
  - Analyzing CI security and safety is not feasible without taking these interconnections into consideration

CISA* list of critical infrastructures (ABCD order)

- Chemical
- Commercial facilities
- Communications
- Critical manufacturing
- Dams
- Defense industrial base
- Emergency services (police, firefighters & similar)
- Energy
- Financial services
- Food and agriculture
- Government facilities
- Information technology
- Healthcare and public health
- Nuclear reactors, materials and waste
- Transportation
- Water and wastewater systems

* CISA = Cybersecurity & Infrastructure Security Agency (CISA) of the USA

ENISA list of critical infrastructures (NIS2)

Annex I

- Energy
- Transport
- Banks
- Financial services
- Health
- Drinking water
- Wastewater
- Digital infrastructure
- ICT service management
- Public infrastructure
- Space

Annex II

- Postal and courier services
- Waste management
- Chemicals
- Food
- Manufacturing
- Digital services
- Research

- Education???
- Note: sectors marked purple not included in the DHS/CISA list

LEGAL FRAMEWORK AND STANDARDS

NIS2 in brief

- **DEF:** The Directive on measures for a high common level of cybersecurity across the Union (NIS2 Directive) of the EU (2022/2555) is a legislative framework with the main goal is to protect critical infrastructures in the EU from cyber attacks
  - Adopted by the Council of Europe: November 10, 2022
  - Entered into force: January 16, 2023
  - Predated by: Directive (EU) 2016/1148 of the European Parliament and of the Council (NIS Directive) → Lower number of sectors and affected entitities

- **DEF:** Act XXIII of 2023 on cybersecurity certification and cybersecurity supervision
  - Entered into force: May 23, 2023 → Overhauled on December 17, 2024
  - Registration deadline: June 30, 2024 → With the relevant authority (Szabályozott Tevékenységek Felügyeleti Hatósága (SZTFH))
  - Compliance deadline: October 28, 2024 → Risk-aligned information security management system (ISMS) → Not just documentation (!), but people, processes and technologies (PPT)
  - Auditor contract: October 28, 2024 (or 120 days after registration)
  - Independent audit: December 31, 2025 → Bi-annual third-party auditing (extra costs!)

How does NIS2 impact the state?

- Translate NIS2 into national legislation → Not just within the EU (!)
- Identify highly critical (Annex I) and other (Annex II) critical infrastructures
- Standardization of cybersecurity controls and services in accordance with Directive EU 2019/881 → Long list of security controls applicable in different criticality levels
- Set up an audit process
- Provide training and support where able (e.g., EDIH projects with state and EU funding)
- Name or found a regulatory body → Inspectors will be necessary as well
- Spill-down impacts to other bodies and companies if they are in the supply chain of critical infrastructures (!)

https://blog.sekoia.io/navigating-the-nis2-directive-key-insights-for-cybersecurity-compliance-and-how-sekoia-io-can-help/

CC in a nutshell

- Common Criteria for Information Technology Security Evaluation
  - Common Criteria or CC
- International standard in **computer security certification** (ISO/IEC 15408)
- Allows vendors (i.e., producers of computer hardware & software) to specify their Security Target (ST) as a function of
  - Security Functional Requirements (SFR) and
  - Security Assurance Requirements (SAR).
- Vendors can implement what they specify, and security testing laboratories evaluate those claims.
- This security evaluation ensures that the overall security maturity of the product is in line with the security requirements of target deployment environments.
- **NOTE:** The CC becomes highly relevant with NIS2 which includes security certification (!)

NIST framework

- When: February 2014 (v1.0) → followed up by v1.1 in April 2018
  - Developed after the Cybersecurity Enhancement Act (2014) tasked the NIST to develop a voluntary framework
- What: a cybersecurity risk framework for voluntary use by critical infrastructure owners and operators
- Where: available online freely https://nvlpubs.nist.gov/nistpubs/cswp/nist.cswp.04162018.pdf
- Consists of three elements:
  - The **Framework Core** is a set of cybersecurity activities, outcomes, and informative references that are common across sectors and critical infrastructure
  - **Framework Profiles** help an organization to align and prioritize its cybersecurity activities with its business/mission requirements, risk tolerances, and resources
  - **Framework Tiers** provide a mechanism for organizations to view and understand the characteristics of their approach to managing cybersecurity risk

NIST 800-53

- Definition: Catalog of security controls for all U.S. federal information systems
  - except those related to national security
- Goal(s):
  - to assist federal agencies in implementing the Federal Information Security Management Act of 2002 (FISMA) and
  - to help with managing cost effective programs to protect their information and information systems.

- AC – Access Control
- AT – Awareness and Training
- AU – Audit and Accountability
- CA – Security Assessment and Authorization
- CM – Configuration Management
- CP – Contingency Planning
- IA – Identification and Authentication
- IR – Incident Response
- MA – Maintenance
- MP – Media Protection
- PE – Physical and Environmental Protection
- PL – Planning
- PS – Personnel Security
- RA – Risk Assessment
- SA – System and Services Acquisition
- SC – System and Communication Protection
- SI – System and Information Integrity
- PM – Program Management

NIST Special Publication (SP) 800-82

- Title: Guide to Industrial Control Systems (ICS) Security
- Revision 2, May 2015
- Guidance for securing Industrial Control Systems (ICS), including
  - Supervisory Control and Data Acquisition (SCADA)
  - Distributed Control Systems (DCS)
  - Programmable Logic Controllers (PLC)
- "*…an overview of ICS and typical system topologies, identifies typical threats and vulnerabilities to these systems, and provides recommended security countermeasures to mitigate the associated risks.*"

National Institute of Standards and Technology
U.S. Department of Commerce

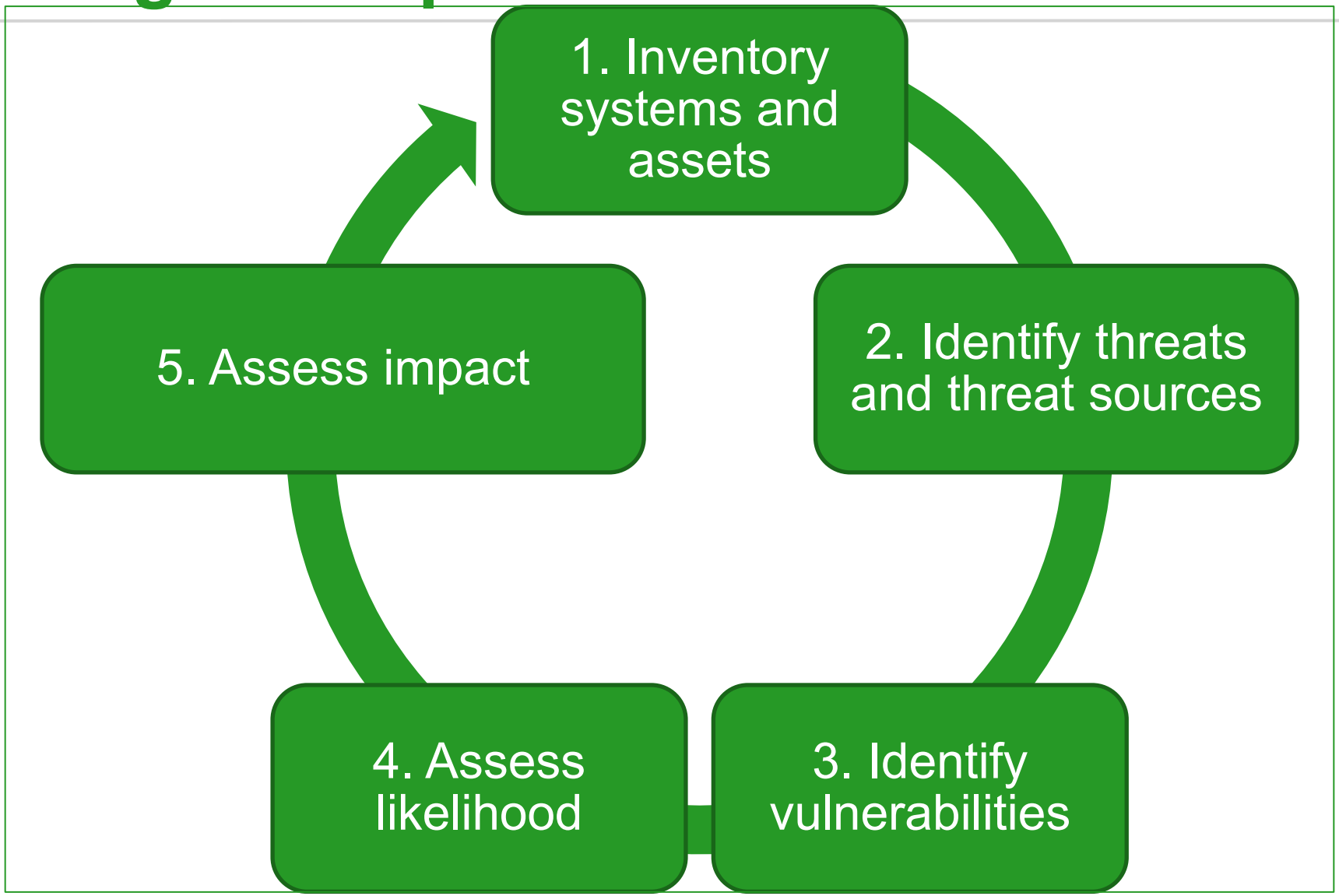
NERC CIP Overview

- North American Electric Reliability Corporation (NERC) issues the Critical Infrastructure Protection (CIP) suite of binding regulations to energy-sector actors

- CIP002-5.1: Critical Cyber Asset Identification
- CIP003-6: Security Management Controls
- CIP004-6: Personnel & Training
- CIP005-5: Electronic Security Perimeter(s)
- CIP006-6: Physical Security of Critical Cyber Assets
- CIP007-6: Systems Security Management
- CIP008-5: Incident Reporting and Response Planning
- CIP009-6: Recovery Plans for Critical Cyber Assets
- CIP010-2: Configuration Change Management and Vulnerability Assessments
- CIP011-2: Information Protection
- CIP013-1: Supply Chain Risk Management (draft – maybe in v7)
- CIP014-2: Physical Security

Risk Management

Risk management process



(1) Inventory of endpoints and key assets

- Endpoints might provide access to adversaries (beside regular uses)
- Physical access points (PAP)
  - Building/room entrances
  - People
  - Workstations & mobile devices
  - Remote equipment
- Electronic access points (EAP)
  - Web presence
  - Business services e.g., APIs
  - Mobile devices
- Risk management starts with ensuring an inventory of key assets and endpoints exists
- Adversaries will look for vulnerabilities in endpoints to obtain access to key assets

(2) Identification of threat sources

Adversary	Description (as defined in NIST IR 7628)
Nation states	State-run, well organized and financed. Use foreign service agents to gather classified or critical information from countries viewed as hostile or as having an economic, military or a political advantage.
Hackers	A group of individuals (e.g., hackers, phreakers, crackers, trashers, and pirates) who attack networks and systems seeking to exploit the vulnerabilities in operating systems or other flaws.
Terrorists / Cyberterrorists	Individuals or groups operating domestically or internationally who represent various terrorist or extremist groups that use violence or the threat of violence to incite fear with the intention of coercing or intimidating governments or societies into succumbing to their demands.
Organized Crime	Coordinated criminal activities including cyber extortion, IP theft and other. An organized and well-financed criminal organization.
Other Criminal Elements	Another facet of the criminal community, which is normally not well organized or financed. Normally consists of few individuals, or of one individual acting alone.
Industrial Competitors	Foreign and domestic corporations operating in a competitive market and often engaged in the illegal gathering of information from competitors or foreign governments in the form of corporate espionage.
Disgruntled Employees	Angry, dissatisfied individuals with the potential to inflict harm on critical infrastructures. This can represent an insider threat depending on the current state of the individual's employment and access to the systems.
Careless or Poorly Trained Employees	Those users who, either through lack of training, lack of concern, or lack of attentiveness pose a threat to CI. This is another example of an insider threat or adversary.

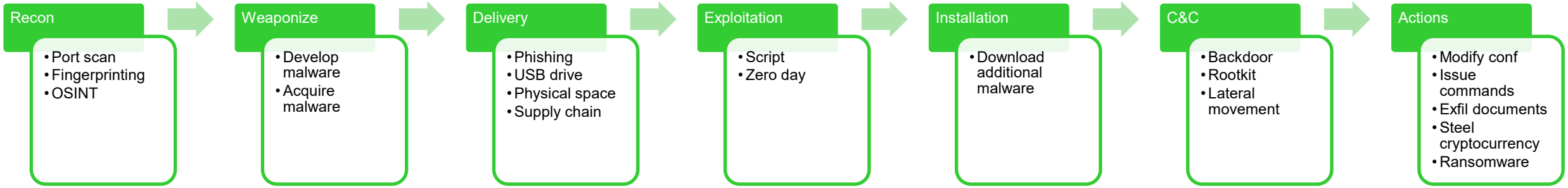
Note: Careless or poorly trained employees are actually vulnerabilities not threat sources (!)

Risk Analysis - Threats

- The most likely threat sources targeting CI are
  - insiders,
  - actors sponsored by Advanced Persistent Threats (APT), or
  - professional hacker groups which carry out attacks
- Common motivation: some form of financial reward e.g., ransomware campaigns, terrorism (Ukraine 2015, ~~Stuxnet~~).

Risk Analysis – Threat actor operations (2024)

- The usual modus operandi (operations) of cyber adversaries are described below (2024).
  - Not a waterfall process: Additional recon might be necessary even after installation!
- It is important to be aware of cyber adversary capabilities as they influence the risk assessment (both likelihood & impact) process.



(3) Assess likelihood

- After clearly defining the threat sources and threats, the next step of the methodology is to determine the probability of their realization
- NIST suggests five levels of probability that a particular threat will materialize:
  - 1. Very High: almost certain that the given threat will be realized
  - 2. High
  - 3. Moderate
  - 4. Low
  - 5. Very Low: almost non-existent or very unlikely
- Likelihood assessment can be based on sector-specific publications and shared experiences
- Note: The highest level should be chosen if both motivated adversaries and known vulnerabilities exist.

Likelihood (alternate view)

- The Energy Community of Europe, which includes the countries of the European Union and a number of other countries, presents the probability of realization of potential threats through 4 levels:

Likelihood level	Description (Likelihood level assigned if one of criteria met)	Indicative frequency (expected)
Almost certainly	The stakeholder is exposed to this threat on a daily basis; common threats (e.g. malicious code).	daily
Probably	The stakeholder is exposed to this threat several times a year; incidents based on this threat occurs in the country regularly (e.g. on monthly basis)	monthly
Possibly	The stakeholder is exposed to this threat on a yearly basis; incidents based on this threat occurs in the country on regularly (e.g. several times a year); such events occurs in the region on monthly basis	yearly
Rarely	The stakeholder is exposed to this threat; incidents based on this threat occurred in the country; such events occurred in the region in the last years	Once in several years

(4) Assess impact (NIST view)

- Impact types (according to NIST)
  - the proper functioning of the system,
  - financial losses,
  - penalties by regulatory agencies,
  - (physical) infrastructure damage,
  - impact on other organizations,
  - company reputation,
  - personal data breaches,
  - and ultimately personal injury and the loss of human lives

- Depending on impact(s), the qualitative impact levels can be categorised into:
  - 1. Very High
  - 2. High
  - 3. Moderate
  - 4. Low
  - 5. Very Low

EU view of impact

- The categories for assessing and quantifying the impacts on the relevant foreseen categories in EU legislation are:
  - Impacts on people (usually measured by numbers)
  - Economic and environmental impacts (usually measured in euros)
  - Political / social influences (hard to quantify, usually qualitatively-measured)
- Semi-quantitative **impact scale** with the following levels:
  - Catastrophic
  - Significant
  - Moderate
  - Minor

(5) Evaluate risk with the risk matrix (NIST)

- NIST Assessment Scale – Level Of Risk (Combination of Likelihood and Impact)

Likelihood (Threat Event Occurs and Results in Adverse Impact)	Level of Impact				
	Very Low	Low	Moderate	High	Very High
Very High	Very Low	Low	Moderate	High	Very High
High	Very Low	Low	Moderate	High	Very High
Moderate	Very Low	Low	Moderate	Moderate	High
Low	Very Low	Low	Low	Low	Moderate
Very Low	Very Low	Very Low	Very Low	Low	Low

Assessment Scale

- NIST Assessment Scale – Level Of Risk

Qualitative Values	Semi-Quantitative Values		Description
Very High	96-100	10	**Very high risk** means that a threat event could be expected to have **multiple severe or catastrophic** adverse effects on organizational operations, organizational assets, individuals, other organizations, or the Nation.
High	80-95	8	**High risk** means that a threat event could be expected to have a **severe or catastrophic** adverse effect on organizational operations, organizational assets, individuals, other organizations, or the Nation.
Moderate	21-79	5	**Moderate risk** means that a threat event could be expected to have a **serious** adverse effect on organizational operations, organizational assets, individuals, other organizations, or the Nation.
Low	5-20	2	**Low risk** means that a threat event could be expected to have a **limited** adverse effect on organizational operations, organizational assets, individuals, other organizations, or the Nation.
Very Low	0-4	0	**Very low risk** means that a threat event could be expected to have a **negligible** adverse effect on organizational operations, organizational assets, individuals, other organizations, or the Nation.

Assessment Scale (EU)

- Risk evaluation level (EU) is defined on matrix function that maps assessed impact consequence (from threats valorized through vulnerabilities) and impact likelihood (the possibility of such threat to occur) to risk level:
  - Risk map (Consequence = Impact, Likelihood = Probability) -> Risk level

Likelihood / Consequence	Rarely	Possibly	Probably	Almost certainly
Catastrophic/ Disastrous	Very High	Very High	Very High	Very High
Significant/Very serious	High	Very High	Very High	Very High
Moderate/Serious	Medium	High	High	High
Minor/ Substantial	Low	Medium	Medium	Medium

Discuss next steps

- Example risk identified during the risk management process: Competitor from a foreign country breaches R&D system, steals secret production recipe for a new medication, starts production and enters earlier.
  - Likelihood: High
  - Impact: High

- Discussion:
  - Why are both likelihood and impact high?
  - What next?
  - Which plans to develop?

CAREER PATHS IN CYBER SECURITY

Introduction

Probable career pathways

- Most job ads need prior experience
- Job offers are usually too generic and wide → Employers attempt to reach an as wide as possible pool of talent
- Job roles (very) broadly separable
  - Management, legal and audit
  - Technical – developer / implementer
  - Technical – response
  - Tester
  - Data-intensive
- Participation in cyber security competitions are excellent references

ENISA skills framework (2022)

- Chief Information Security Officer (CISO)
- Cyber Incident Responder
- Cyber Legal, Policy & Compliance Officer
- Cyber Threat Intelligence Specialist
- Cybersecurity Architect
- Cybersecurity Auditor
- Cybersecurity Educator
- Cybersecurity Implementer
- Cybersecurity Researcher
- Cybersecurity Risk Manager
- Digital Forensics Investigator
- Penetration Tester

Ethical hacking: CEH

Category	Description
Title	Certified Ethical Hacker
Issued by & year	EC-Council, 2003
Topics	20 modules: scanning & enumeration, social engineering, session hijacking, hacking web servers/applications, cloud, web, OT, WPA3, enumeration, reverse engineering, etc.
Certification cost(s)	500+ USD
Website	https://www.eccouncil.org/programs/certified-ethical-hacker-ceh/
Training(s)	EC-Council (self-study, classroom)
Book(s)	

Ethical hacking: OSCP

Category	Description
Title	Offensive Security Certified Professional
Issued by & year	Offensive Security (OffSec). 25,000+ certificate holders.
Topics	Penetration testing – hack 5 systems in 24 hours
Certification cost(s)	~1800 USD (no renewal necessary)
Website	https://www.offensive-security.com/oscp
Training(s)	OffSec course bundled with exam voucher
Book(s)	

Management and audit: CISSP

Category	Description
Title	Certified Information System Security Professional
Issued by & since	ISC2, 1994. Note: 142000+ certification holders
Topics	risk assessment, asset security, architecture & engineering, communication & network, IAM, assessment & testing, operations, software development
Certification cost(s)	700+ USD (3-year validity → renewal or CPE + fee)
Website	https://www.isc2.org/Certifications/CISSP
Training(s)	ISC2, Udemy
Book(s)	Different CISSP exam prep guides – usually too big!

Management and audit: CISM

Category	Description
Title	Certified Information Security Manager
Issued by & since	ISACA, 2002
Topics	governance, risk management, program, incident management
Certification cost(s)	575 USD (CPE credits to maintain)
Website	https://www.isaca.org/credentialing/cism
Training(s)	ISACA (~1000 USD), Udemy (85 USD)
Book(s)	

Management and audit: CISA

Category	Description
Title	Certified Information **Systems** Auditor
Issued by & since	ISACA, 1978
Topics	Auditing process, governance & management of IT, acquisition-development-implementation, operations + business resilience, protection of information assets
Certification cost(s)	575 USD
Website	https://www.isaca.org/credentialing/cisa
Training(s)	ISACA (895 USD), Udemy (75 USD)
Book(s)	

Operational technology: GICSP

Category	Description
Title	Global Industrial Cyber Security Professional (GICSP)
Issued by & since	GIAC/SANS,
Topics	components, purposes, deployments, attach surfaces, incident response, governance
Certification cost(s)	1899 USD + 8040 USD for the course (SANS)
Website	https://www.giac.org/certifications/global-industrial-cyber-security-professional-gicsp/
Training(s)	ICS410: ICS/SCADA Security Essentials (course), 36 CPE
Book(s)	*NO BOOKS*

Operational technology: CSSA

Category	Description
Title	Certified SCADA Security Architect (CSSA)
Issued by & since	Information Security Institute
Topics	Policy, standards, access control, protocols & field communications, AAA, vulnerability assessments
Certification cost(s)	599 USD (7-day free trial)
Website	https://www.infosecinstitute.com/skills/learning-paths/certified-scada-security-architect-cssa/
Training(s)	InfoSec Institute, Udemy
Book(s)	*NO BOOKS*

Incoming & latest: CCSP

Category	Description
Title	Certified Cloud Security Professional
Issued by & year	ISC2, 2015.
Topics	Architecture & design, cloud data, platform & infrastructure, application, operations, legal-risk-compliance
Certification cost(s)	599 USD
Website	https://www.isc2.org/Certifications/CCSP
Training(s)	ISC2
Book(s)	

Incoming & latest: CSSLP

Category	Description
Title	Certified Secure Software Lifecycle Professional
Issued by & year	ISC2, 2008.
Topics	Requirements, architecture & design, implementation, testing, lifecycle management, deployment & operations, supply chain
Certification cost(s)	599 USD, valid 3 years, 90 CPEs to renew
Website	https://www.isc2.org/Certifications/CSSLP
Training(s)	ISC2
Book(s)	Conklin A., Shoemaker D., “CSSLP Certified Secure Software Lifecycle Professional All-in-One Exam Guide”, McGraw Hill, 2022

Conclusion

- Data privacy in a nutshell
- Critical infrastructure sectors
- Legal framework and standards
- Risk management
  - (Likelihood, Impact) → Risk matrix
- Possible career paths in cybersecurity
  - Hacking, management & incoming

Additional references

- Legal
  - EU GDPR, https://eur-lex.europa.eu/eli/reg/2016/679/oj/eng
  - EU GDPR (more easily readable and linked), https://gdpr-info.eu/
  - EU NIS2, https://eur-lex.europa.eu/eli/dir/2022/2555/2022-12-27/eng
  - Act XXIII, https://njt.hu/jogszabaly/en/2023-23-00-00
- Certification providers
  - SANS/GIAC training & certifications (2025). https://www.sans.org/cyber-security-certifications/
  - ISC2 training & certifications (2025). https://www.isc2.org/certifications
  - OffSec training & certifications (2025). https://www.offsec.com/courses-and-certifications/
  - EC-Council training & certifications (2025). https://www.eccouncil.org/train-certify/

Thank you for your attention!

Eötvös Loránd University (ELTE)
Faculty of Informatics (IK)
Pázmány Péter sétány 1/c
1117 Budapest, Hungary

A BRIEF INTRODUCTION INTO AI SECURITY
- PART 1 -

Introduction to Data Security

Imre Lendák, PhD, GICSP

2026
Budapest, Hungary

Talk overview

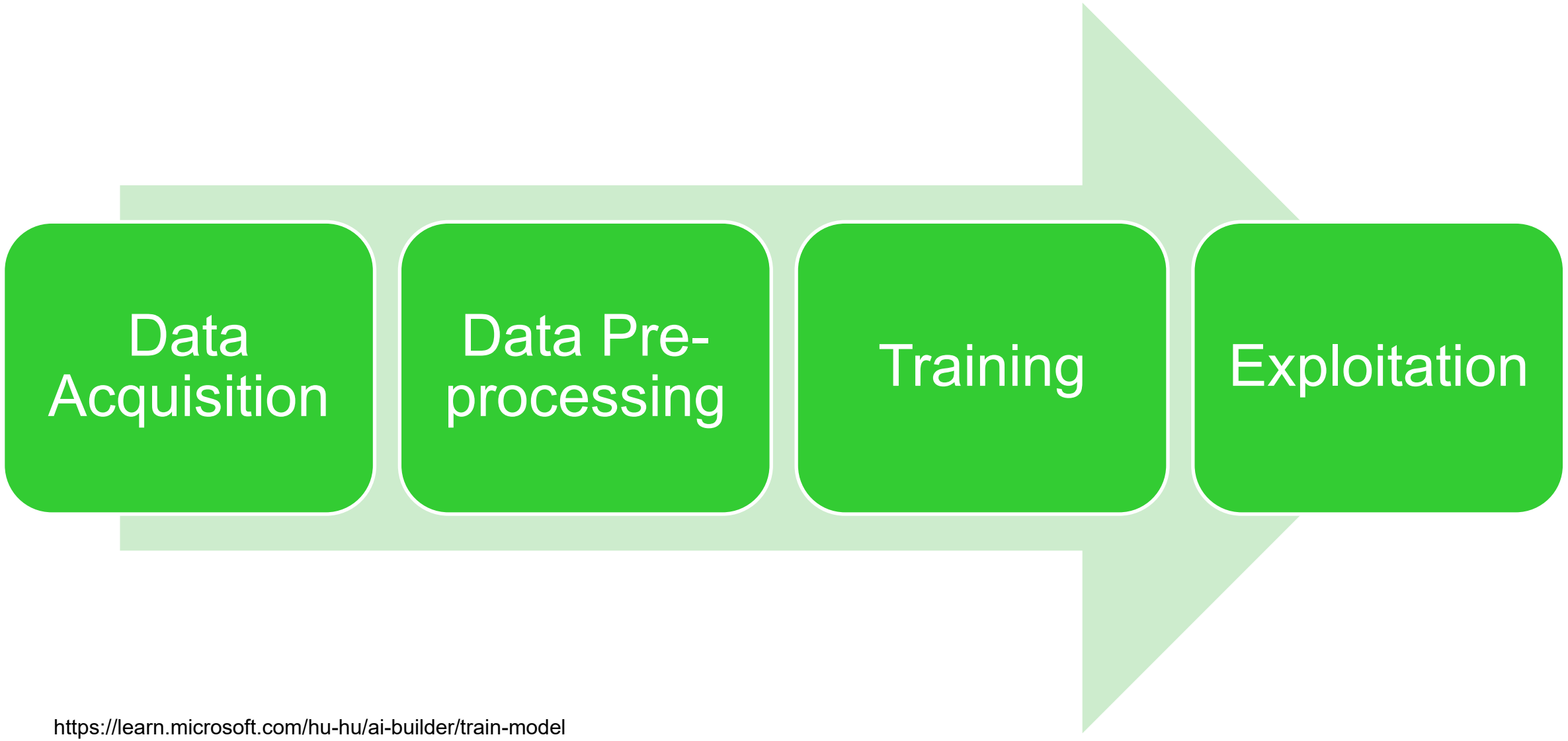
- Brief introduction & motivation
- Security of applied AI
- AI security ontology

Motivation

- Teaching focus:
  - DigitalTech EDIH project (2023-2025) – Micro course on AI security since 2023
  - Data Security course starting in 2024/2025
- Research focus:
  - Synthetic data generation (1 PhD done, 1 underway)
  - Agentic AI security (1 PhD underway)
  - Explainable AI (1 PhD almost done)
  - Federated learning (1 PhD done)
  - AI forensics

SECURITY OF APPLIED AI

AI lifecycle

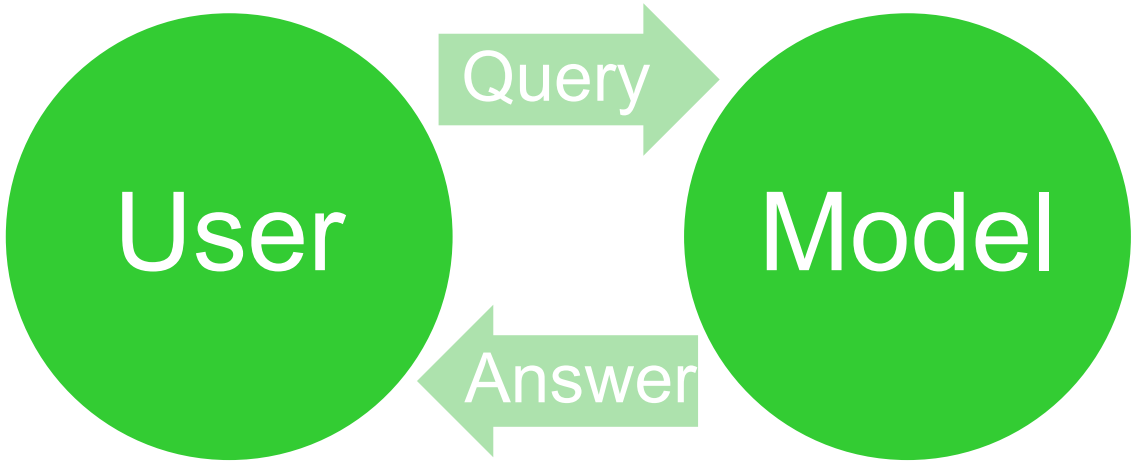


https://learn.microsoft.com/hu-hu/ai-builder/train-model

Training phase simplified + attack surface

- The attacker's motivation is usually data theft, but the model can be targeted as well
  - Data is usually kept secret in certain fields e.g., medicine, finances, energy
  - The algorithm is usually published or mainly public
  - Models can be made public or kept internal

AI exploitation



- The AI model is used to solve a real problem
  - The user formulates a query
  - The model parses the query and formulates an answer
  - The answer is correct or hallucination (?)
- Model access control
  - Internal
  - Public
  - Subscription-based
- Seemingly smaller attack surface compared to the training phase → many more actors can access the AI-based system → more attackers (!)

AI attacks and the CIA triad

C

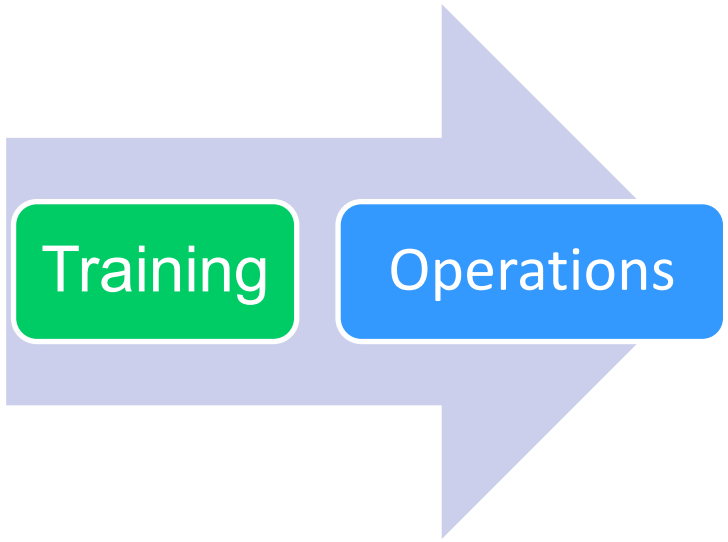
- Data theft
- Model theft
- Algorithm theft

I

- Data poisoning
- Algorithm integ.
- HP integrity

A

- Model poisoning



C

- Mem.inference
- Model inversion

I

- Model poisoning
- Data poisoning

A

- Sponge attacks

Evasion attacks*

Applied AI in critical sectors

- **Transport:** video, audio and sensor data analytics (e.g., in smart vehicles)
- **Healthcare:** video, audio and sensor data analytics (ECG, EEG), statistical data analysis
- **Space:** video, audio and sensor data analytics (e.g., space & other telescope)
- **Science:** statistical data analysis, video, audio and sensor data analytics
- **Finance & banking:** fraud detection, credit rating, price analytics
- **Manufacturing:** real-time monitoring, time-series analysis, predictive maintenance
- **Energy:** load forecasting, predictive maintenance
- **Commercial:** consumption habits, recommender engines

- Common AI tasks: classification, forecasting/regression, clustering, frequent pattern mining
- Common in all or at least many:
  - **Cybersecurity:** avoid unauthorized access and loss of integrity of data and services
  - **Large language models:** crunch and store large volumes of data, reproduce

Classification

- **DEF:** Classification models assign observations to predefined classes
  - Supervised learning → Curated, annotated training datasets
- Hyperparameters (algo & model-agnostic):
  - Distance metric
  - Distance threshold
  - Sensitivity → Low false negatives (FN)
  - Specificity → Low false positives (FP)
  - Precision → $\frac{(TP+TN)}{(TP+FP+TN+FN)}$
- **Note:** Most evasion attacks target and evade classification models

- Possible attacker goals:
  - Class-generic attack → Maximize the misclassification rate, regardless of class
  - Class-specific attack → Maximize probability of misclassification into specific class

Anomaly detection

- **DEF:** The goal of anomaly (or outlier) detection is to identify observations which are significantly different
  - **DEF:** In a **point anomaly** an individual data instance is anomalous with respect to its surroundings
  - **DEF:** In **contextual anomalies** a data instance is anomalous in specific context
  - **DEF: Collective anomalies** are collections of observations anomalous in relation to the entire data set
  - **Note:** Anomaly detection is often treated as binary classification (anomaly YES/NO)
- Attacker goal: Camouflage suspicious activity or content as legitimate
- Change detection (CD) != anomaly detection (AD)
  - Preventive maintenance in manufacturing → Identify minute (i.e., very small) changes in sensor data → Declare change → Perform preventive maintenance → Avoid costly equipment failure

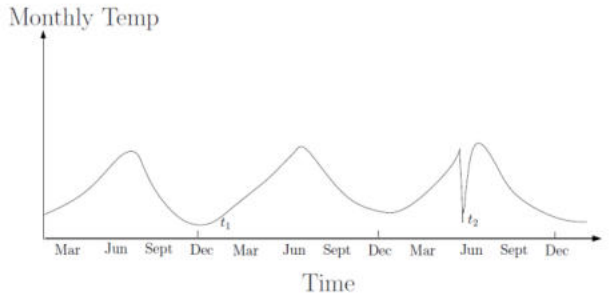


Fig. 3. Contextual anomaly t_2 in a temperature time series. Note that the temperature at time t_1 is same as that at time t_2 but occurs in a different context and hence is not considered as an anomaly.

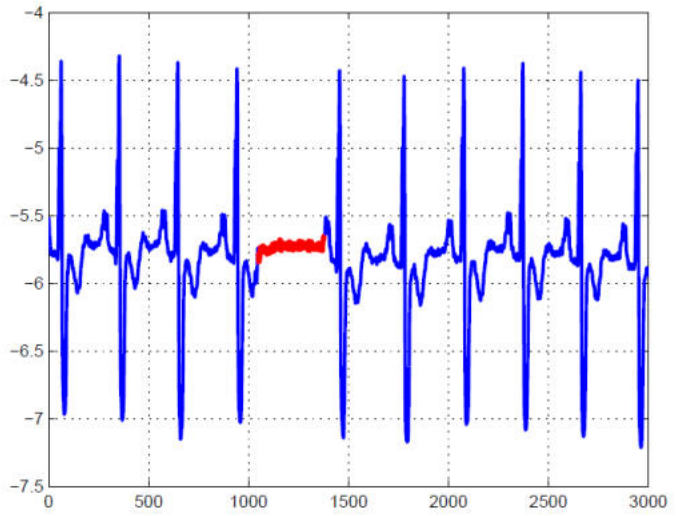


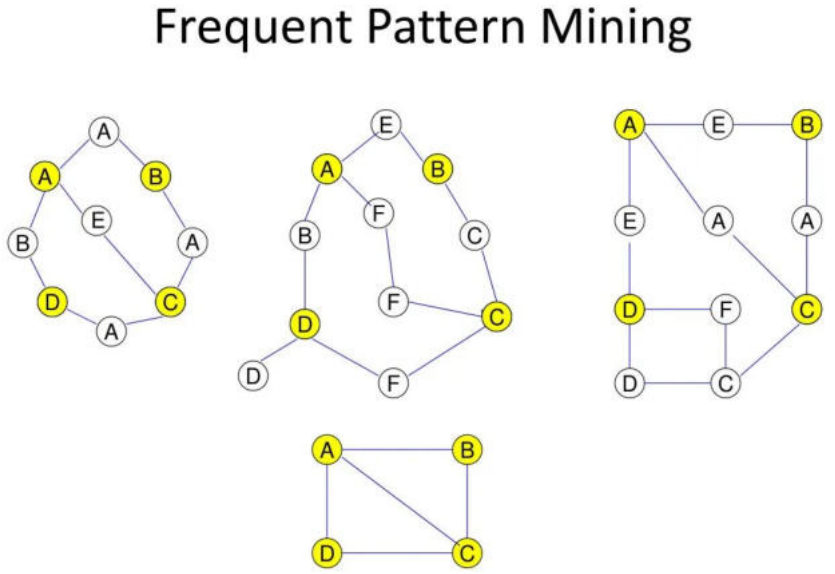
Fig. 4. Collective anomaly corresponding to an *Atrial Premature Contraction* in an human electrocardiogram output.

Clustering

- **DEF:** The goal of clustering is to group similar observations into clusters
  - Unsupervised learning → Data observations are not annotated upfront
- The training algorithm maximizes:
  - Intra-cluster homogeneity
  - Inter-cluster distance
- Hyper parameters (algo & model-agnostic)
  - Distance metric
  - Similarity threshold
  - Number of clusters
  - Max or min number of observations (in clusters)
  - Outlier approach (not really a HP)
- Possible attacker goals:
  - Alter clustering if results are used in downstream classification tasks
  - Insert outliers to distort cluster shapes or impact algorithm convergence

Pattern mining and association rules

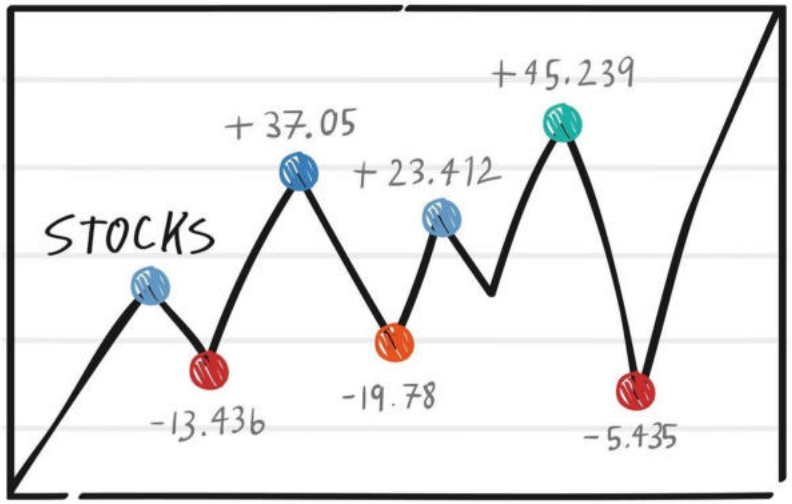
- **DEF:** The goal of pattern mining algorithms is to analyze elements which frequently appear together
- Learning association rules: $[X_1, X_2, \dots, X_n] \rightarrow Y$
  - If certain elements are within a set, then another element will be there with high probability as well
- **DEF:** The goal of sequential pattern mining is to identify observations which occur in certain sequence → $[X_1, X_2, \dots, X_n,] + [Z_1, Z_2, \dots, Z_n,] + [\ \dots\ ] \rightarrow Y$
  - Note: The temporal dimension is significant
- Hyper parameters:
  - Maximum pattern/rule length
  - Support: how often does a certain pattern appear → How well is it supported by actual observed data?
  - Confidence: Likelihood of observation Y appearing after X (or sequence ABC) occurred.
  - Lift: The ratio between observed and expected support



https://khadkagopal.medium.com/mining-frequent-pattern-and-association-dffead065ca5

- Possible attacker goals:
  - Slightly alter observations to avoid frequency or sequence threshold triggering

Regression



- **DEF:** The goal of regression algorithms is to fit an optimal function to the observations
  - Plot a best fit curve between the data points
  - Predict the future values of
  - Analyze the impact of variables
- Hyper parameters:
  - Function type e.g., polynomial, shape in N dim space
  - Function parameters
- Possible attacker goals:
  - Poison training data or model to negatively impact function fitting task
  - Alter input data to cause incorrect future predictions

AI tasks and attacks

Classification

- Evasion;
- Model theft
- Data leak

Clustering

- Data poisoning

Regression

- Data leak

Pattern mining

- Data leak

Generative AI

- Prompt injection
- Data leak
- Hallucinations

Agentic AI

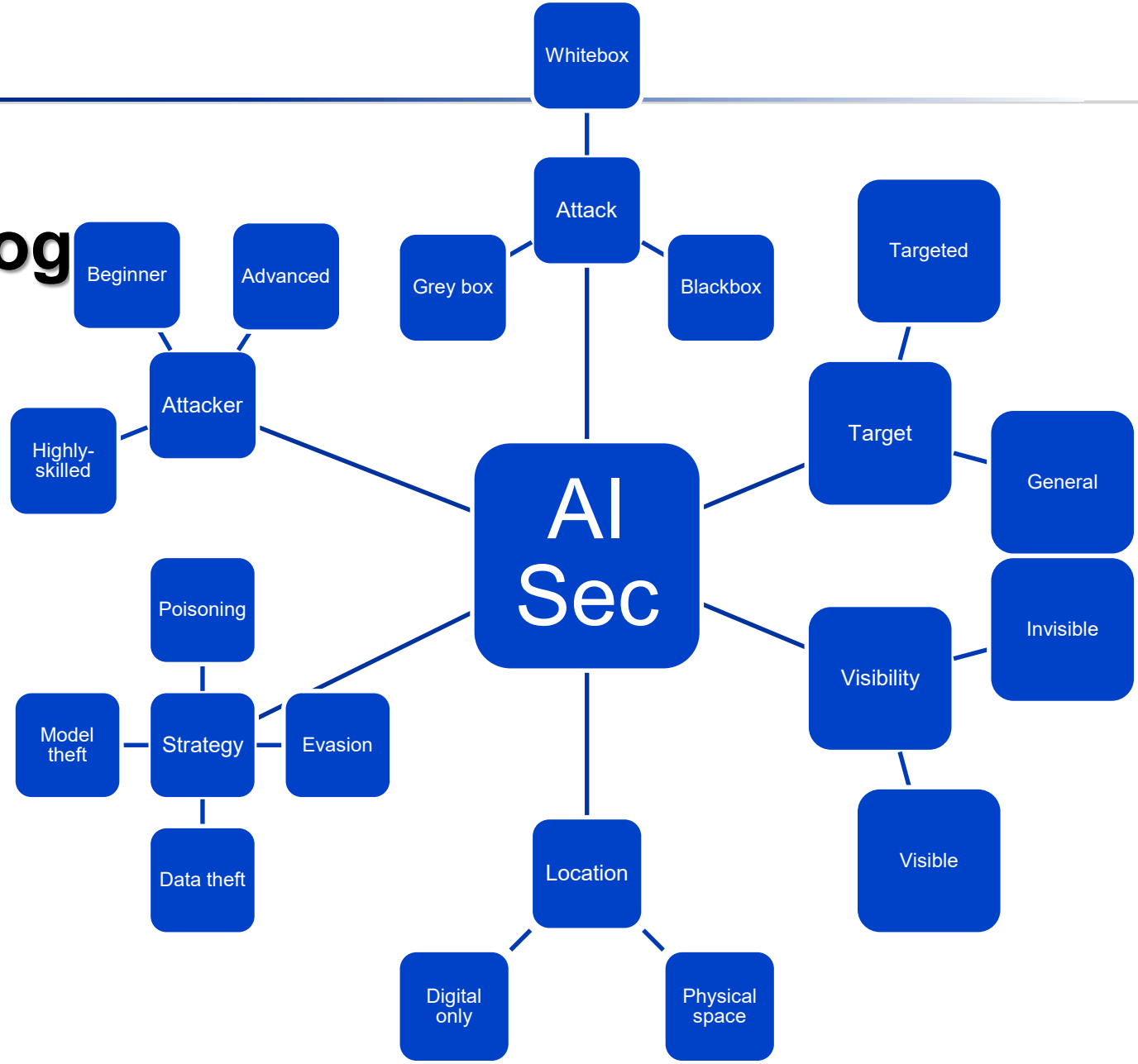
- Excessive agency
- Data leak

AI tasks and attacks

AI task	Evasion	Poisoning*	Data leak	Model theft
Classification	Yes	Maybe	Yes	Yes
Clustering		Yes		
Regression	Maybe	Maybe	Maybe	Yes
Association mining		Maybe	Yes	Maybe
Generative AI		Yes	Yes	Yes
Agentic AI		Yes	Yes	Maybe

- Discuss:
  - Why are attackers motivated to steel classification, regression or generative AI models?
  - Why are evasion attacks mostly linked to classification and (maybe) regression models?
  - How to do poisoning attacks against various AI models?
  - What kind of data might leak from classification, association mining, generative AI or (maybe) regression models? → Link this discussion to the type of data used in the training phase (!)

AI Security Ontolog



Attacker and attack types – A brief overview

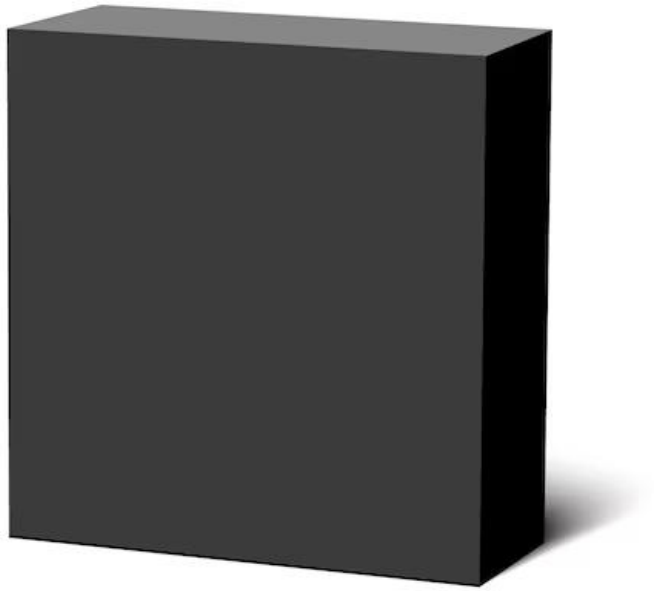
- **Attack types** based on attacker knowledge:
  - Blackbox: No access to training data & model
  - Gray box: Limited training data and/or model access
  - Whitebox: Training data and model known to attacker(s)
- Attacker types based on skills and information collection capabilities:
  - Beginner
  - Advanced
  - Highly skilled

https://www.freepik.com/vectors/cyber-terrorism

AI ATTACK TYPES

Blackbox attacks

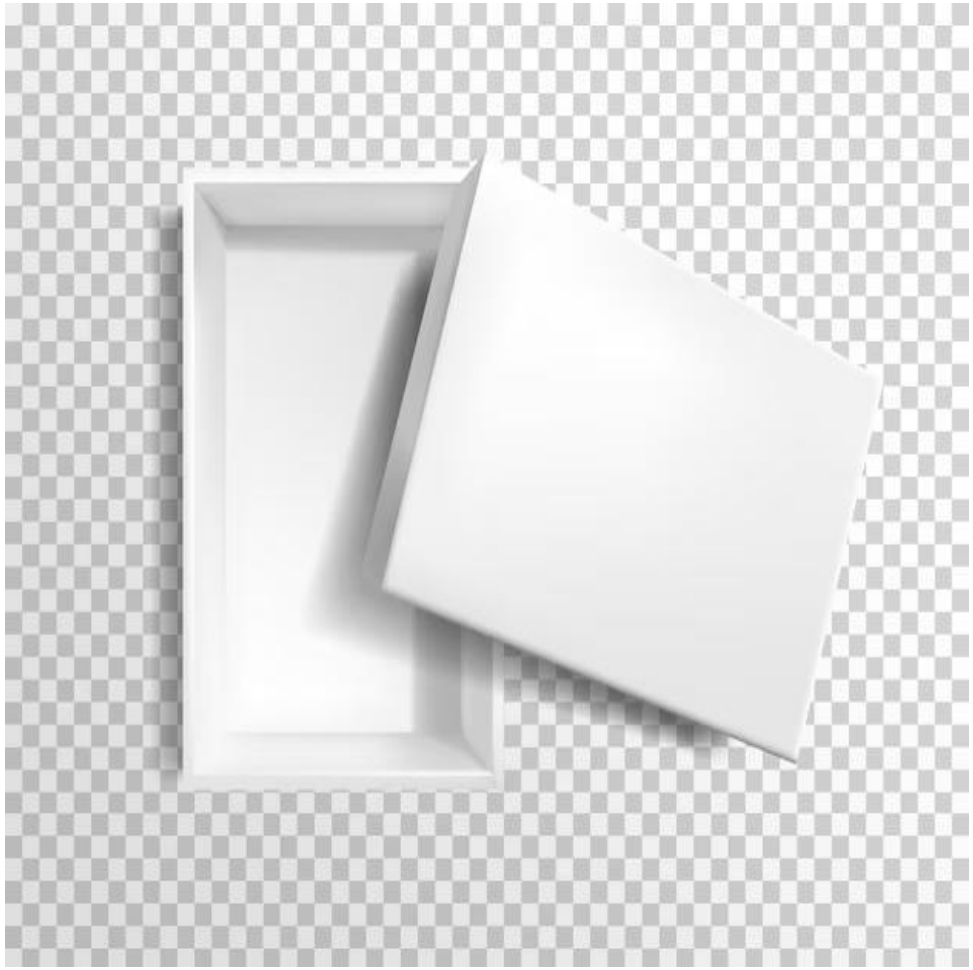
- **Definition:** The attacker does not have access to (training data, algorithm, model architecture, model, hyper parameters) tuple, but is able to formulate valid queries and analyse answers
- **Access:** via API or web interface → query number limiting is important (!)
- **Threat source:** anybody with access to API or model
- **Attack types:** inversion, model poisoning
- **Method:** numerous (query, answer) pair acquisition and analysis
- **Likelihood:** high
- **Discuss:** Link to training & exploitation!



https://www.freepik.com

Whitebox attacks

- **Definition:** The attacker knows the (training data, algorithm, model architecture, model, hyper parameters) tuple
- **Threat source:** malicious insider, model inversion
- **Access:** full
- **Attack types:** (internal) vulnerability analysis, evasion data, training data inversion (if not known)
- **Likelihood:** low



https://www.freepik.com

Gray box attacks

- **Definition:** The attacker has partial access to the (training data, algorithm, model architecture, model, hyper parameters) tuple
- **Information sources:** model inversion, public info (algo, data, architecture), insider information
- **Access:** API or web interface
- **Attack types:** penetration testing/vulnerability analysis, model inversion, design evasion attacks
- **Likelihood:** low to medium

https://www.freepik.com

AI ATTACKER TYPES

Beginner

- Characteristics:
  - Training data access: usually full (e.g. public data).
  - Training data distribution: limited.
  - Training algorithm: limited.
  - AI model architecture: limited.
  - Resources: minimum e.g., personal computer (own laptop).
- Methods applied:
  - Model poisoning: random values and annotations in classification tasks.
  - Denial of service: API or web interface overloading.
  - Input data modification: basic evasion (?)

https://www.freepik.com

Advanced

- Characteristics:
  - Training data access: full (e.g. public data or insider).
  - Training data distribution: full.
  - Training algorithm: advanced.
  - AI model architecture: medium e.g., decision boundary knowledge is limited.
  - Resources: medium e.g., public cloud subscription.
- Methods applied:
  - Model inversion and transfer learning.
  - Model poisoning.
  - Evasion attacks: medium quality adversarial data based on training data knowledge.

https://www.freepik.com

Highly skilled/sophisticated attacker

- Characteristics:
  - Training data access: full (e.g. public data or insider).
  - Training data distribution: full.
  - Training algorithm: advanced.
  - AI model architecture: advanced e.g., decision boundary knowledge is adequate.
  - Resources: significant e.g., own GPU units.
- Methods applied:
  - Insider attack; model inversion; transfer learning.
  - Evasion attack: high quality adversarial data based on training data, algorithm and architecture knowledge.
  - Model poisoning: Targeted modification of decision boundaries.

https://www.freepik.com/vectors/cyber-terrorism

Exercise #1: AI attacker type comparison

- 3 attacker types: beginner, advanced, highly skilled
- 3 attack types: white-, black- and gray box attacks
- Attacker knowledge: training/test data, (training) algorithm, AI system architecture & hyper parameters
- AI lifecycle stage: data acquisition and pre-processing, training, exploitation

- **Task:** Use an A4 paper and draw a 2D ontology of the above-listed topics

Labor #1: Attacker type comparison

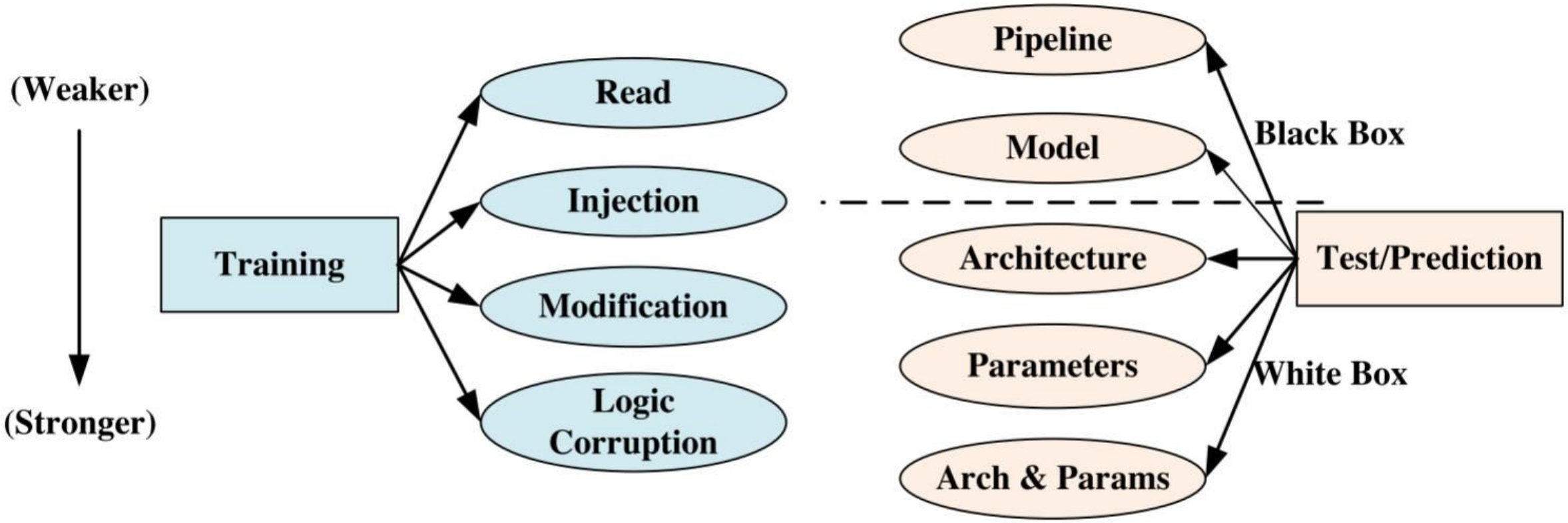


Fig. 5. Adversarial capability [64].

Summary

- Brief introduction
- Attack types
  - White → Gray → Blackbox
- Attacker types
  - Beginner → Advanced → Highly skilled

Popular references, courses & Github projects

- OWASP, “Machine Learning Security Top Ten”, https://owasp.org/www-project-machine-learning-security-top-10/
- Nightfall AI, “AI Security 101”, https://www.nightfall.ai/ai-security-101
- Rod Trent, “Must learn AI security (blog), https://github.com/rod-trent/OpenAISecurity/tree/main/Must_Learn
- Marin Ivezic, “Securing AI”, https://securing.ai
- University of Cagliary, “MISec – Machine Learning Security Seminars” (ELSA EU project), https://pralab.github.io/mlsec/past/
- INFOSEC, “AI Security” on Coursera, more on the legal and ethical side, https://www.coursera.org/learn/ai-security
- Microsoft, “Microsoft Vulnerability Severity and Content Classifications for AI Systems ”, https://www.microsoft.com/en-us/msrc/aibugbar
- Rimmer Vera, “Navigating the Security and Privacy Landscape of Modern AI - Vera Rimmer - NDC Security”, https://www.youtube.com/watch?v=_VWITdRbdTw
- SecML library, https://github.com/pralab/secml
- SecMLTorch library, https://github.com/pralab/secml-torch/tree/main

Scientific references: Evasion attacks

- Jáñez-Martino, F., Alaiz-Rodríguez, R., González-Castro, V., Fidalgo, E., & Alegre, E. (2023). A review of **spam email detection**: analysis of spammer strategies and the dataset shift problem. *Artificial Intelligence Review*, *56*(2), 1145-1173.
- Gupta, S., Raja, K., & Passerone, R. (2024). Visual Prompt Engineering for Enhancing **Facial Recognition Systems Robustness** Against Evasion Attacks. *IEEE Access*.
- Birthriya, S. K., Ahlawat, P., & Jain, A. K. (2025). Detection and Prevention of **Spear Phishing** Attacks: A Comprehensive Survey. *Computers & Security*, 104317.
- Muthalagu, R., Malik, J., & Pawar, P. M. (2025). Detection and prevention of **evasion attacks on machine learning models**. *Expert Systems with Applications*, *266*, 126044.

Scientific references: Model poisoning

- Tian, Z., Cui, L., Liang, J., & Yu, S. (2022). A comprehensive survey on poisoning attacks and countermeasures in machine learning. *ACM Computing Surveys*, *55*(8), 1-35.
- Biggio, B., Nelson, B., & Laskov, P. (2012). Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389*.
- Demontis, A., Melis, M., Pintor, M., Jagielski, M., Biggio, B., Oprea, A., ... & Roli, F. (2019). Why do adversarial attacks transfer? explaining transferability of evasion and poisoning attacks. In *28th USENIX security symposium (USENIX security 19)* (pp. 321-338).

Scientific references: Sponge attacks

- Cinà, A. E., Demontis, A., Biggio, B., Roli, F., & Pelillo, M. (2025). Energy-latency attacks via sponge poisoning. *Information Sciences*, *702*, 121905.
- Shumailov, I., Zhao, Y., Bates, D., Papernot, N., Mullins, R., & Anderson, R. (2021, September). Sponge examples: Energy-latency attacks on neural networks. In *2021 IEEE European symposium on security and privacy (EuroS&P)* (pp. 212-231). IEEE.
- Schoof, C., Koffas, S., Conti, M., & Picek, S. (2024, May). Beyond PhantomSponges: Enhancing Sponge Attack on Object Detection Models. In Proceedings of the 2024 ACM Workshop on Wireless Security and Machine Learning (pp. 14-19).
- Lintelo, J. T., Koffas, S., & Picek, S. (2024). The SkipSponge Attack: Sponge Weight Poisoning of Deep Neural Networks. *arXiv preprint arXiv:2402.06357*.

Scientific references: Model theft & watermarking

- Caviglione, L., Comito, C., Guarascio, M., & Manco, G. (2023). Emerging challenges and perspectives in Deep Learning model security: A brief survey. *Systems and Soft Computing*, *5*, 200050.
- Regazzoni, F., Palmieri, P., Smailbegovic, F., Cammarota, R., & Polian, I. (2021). Protecting artificial intelligence IPs: A survey of **watermarking and fingerprinting** for machine learning. CAAI Transactions on Intelligence Technology, 6(2), 180-191.
- Ye, P. G., Li, Z., Yang, Z., Chen, P., Zhang, Z., Li, N., & Zheng, J. (2025). Periodic watermarking for copyright protection of large language models in cloud computing security. *Computer Standards & Interfaces*, 103983.

Scientific references: Synthetic data generation

- Bauer, A., Trapp, S., Stenger, M., Leppich, R., Kounev, S., Leznik, M., ... & Foster, I. (2024). Comprehensive exploration of synthetic data generation: A survey. *arXiv preprint arXiv:2401.02524*.
- Dankar, F. K., Ibrahim, M. K., & Ismail, L. (2022). A multi-dimensional evaluation of synthetic data generators. *IEEE Access*, *10*, 11147-11158.
- Endres, M., Mannarapotta Venugopal, A., & Tran, T. S. (2022, August). Synthetic data generation: A comparative study. In *Proceedings of the 26th international database engineered applications symposium* (pp. 94-102).
- Galloni, A., Lendák, I., & Horváth, T. (2020, October). A novel evaluation metric for synthetic data generation. In *International Conference on Intelligent Data Engineering and Automated Learning* (pp. 25-34). Cham: Springer International Publishing.
- Livieris, I. E., Alimpertis, N., Domalis, G., & Tsakalidis, D. (2024, June). An evaluation framework for synthetic data generation models. In *IFIP International Conference on Artificial Intelligence Applications and Innovations* (pp. 320-335). Cham: Springer Nature Switzerland.
- Lu, Y., Shen, M., Wang, H., Wang, X., van Rechem, C., Fu, T., & Wei, W. (2023). Machine learning for synthetic data generation: a review. *arXiv preprint arXiv:2302.04062*.

Scientific references – Model inversion

- Song, J., & Namiot, D. (2022, September). A survey of the implementations of model inversion attacks. In *International Conference on Distributed Computer and Communication Networks* (pp. 3-16). Cham: Springer Nature Switzerland.
- Zhou et al (2024), “Model Inversion Attacks: A Survey of Approaches and Countermeasures”, https://arxiv.org/abs/2411.10023
- Dibbo, S. V. (2023, July). Sok: Model inversion attack landscape: Taxonomy, challenges, and future roadmap. In *2023 IEEE 36th Computer Security Foundations Symposium (CSF)* (pp. 439-456). IEEE.
- Fredrikson, M., Jha, S., & Ristenpart, T. (2015, October). Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security* (pp. 1322-1333).
- Mostafa, T., Ibrahem, M. I., & Fouda, M. M. (2024, September). Unraveling Model Inversion Attacks: A Survey of Machine Learning Vulnerabilities. In *2024 2nd International Conference on Artificial Intelligence, Blockchain, and Internet of Things (AIBThings)* (pp. 1-8). IEEE.

Thank you for your attention!

Eötvös Loránd University (ELTE)
Faculty of Informatics (IK)
Pázmány Péter sétány 1/c
1117 Budapest, Hungary

A BRIEF INTRODUCTION INTO AI SECURITY
- PART 2 -

Introduction to Data Security

Imre Lendák, PhD, GICSP

2026
Budapest, Hungary

Talk overview

- OWASP Top 10 Machine Learning
  - Brief description of each risk
- Who is who in AI security?

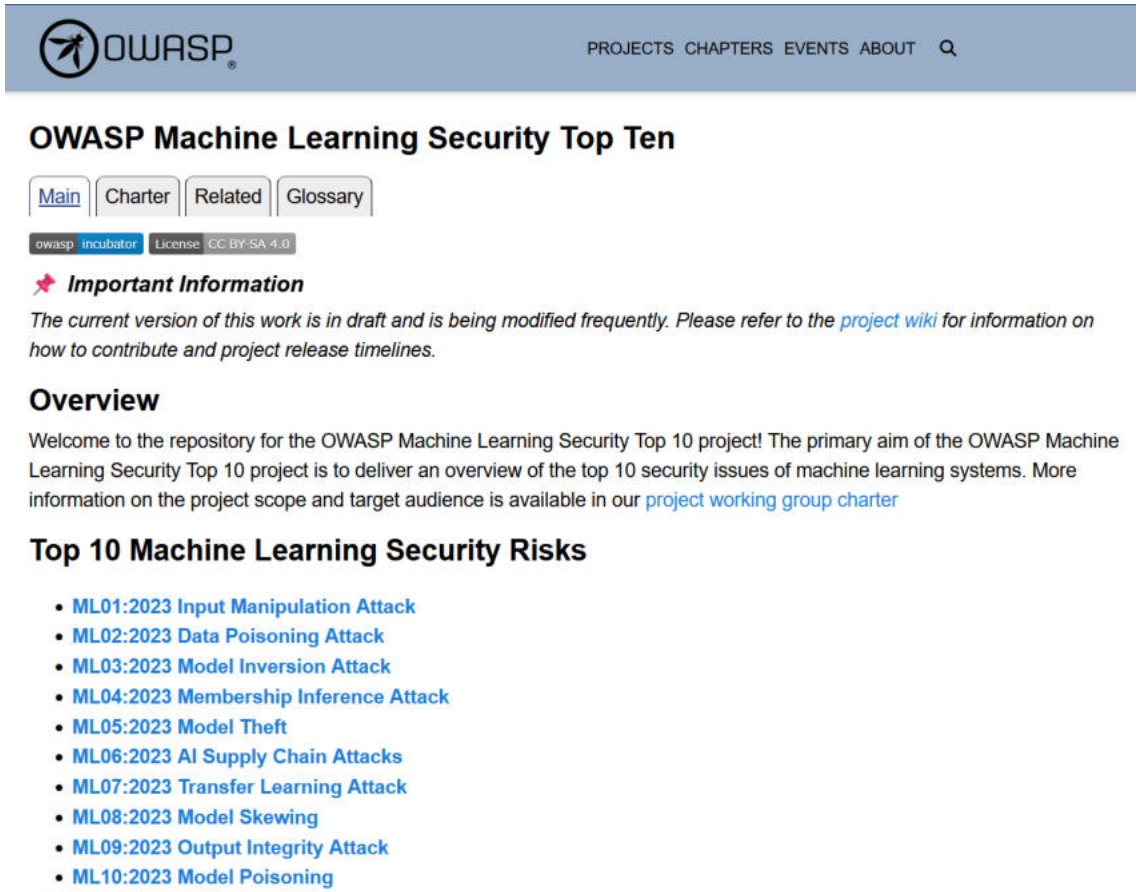
OWASP Top 10

OWASP top 10 lists

- Open Web Application Security Project (OWASP)
- OWASP is an **online community** aiming to improve cyber security in various domains
- The **OWASP Top 10 lists** made them famous
  - The most publicized list is the **OWASP Top 10 for Web Applications**
  - Publish mitigations together with the top vulnerability descriptions
  - The OWASP lists and other materials are **completely free** with open source (Github)
- Other AI-related list(s) : OWASP Top 10 Machine Learning Security + LLM list

OWASP Top 10: Machine Learning Security overview

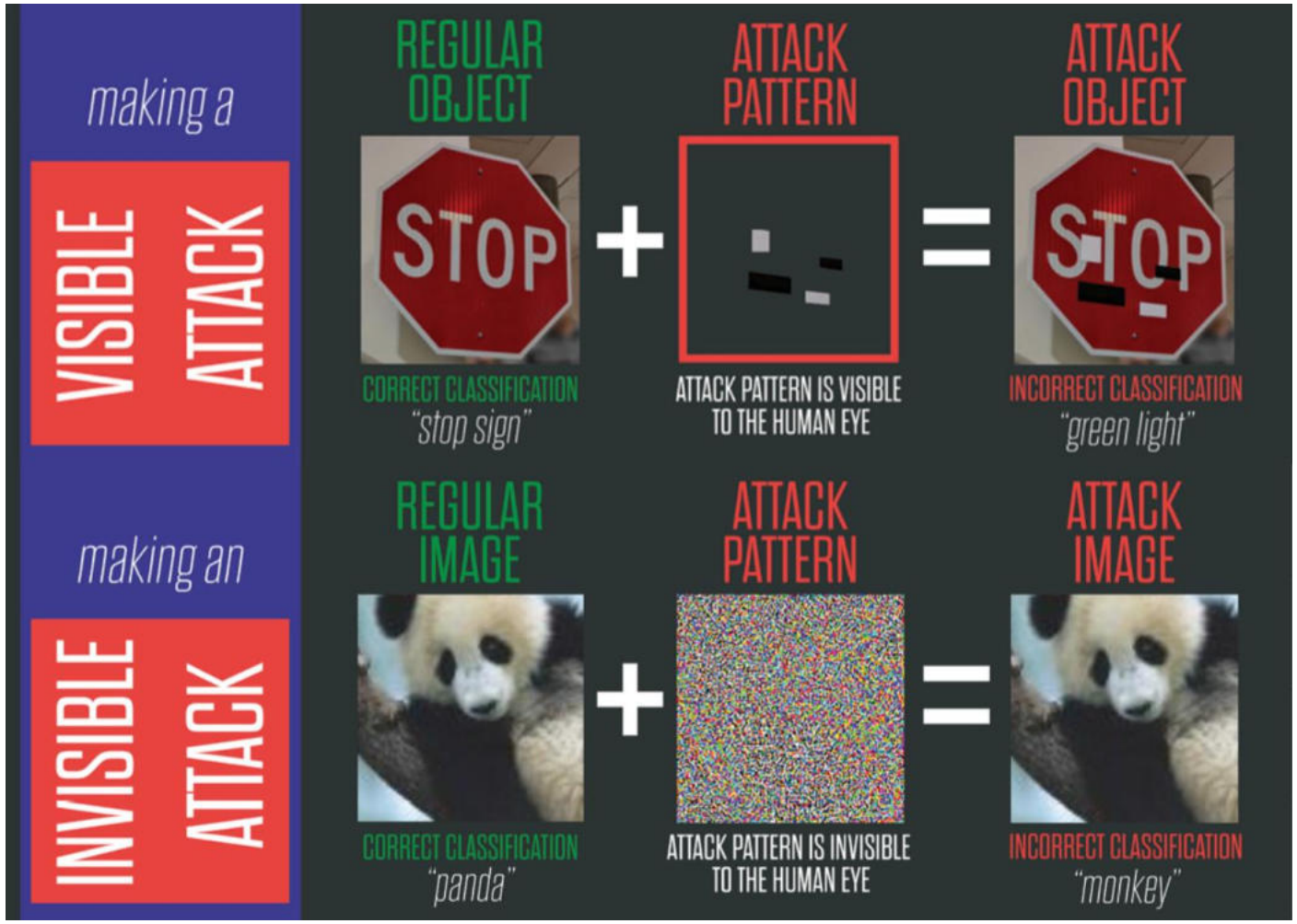
- ML01:2023 Input Manipulation Attack
- ML02:2023 Data Poisoning Attack
- ML03:2023 Model Inversion Attack
- ML04:2023 Membership Inference Attack
- ML05:2023 Model Theft
- ML06:2023 AI Supply Chain Attacks
- ML07:2023 Transfer Learning Attack
- ML08:2023 Model Skewing
- ML09:2023 Output Integrity Attack
- ML10:2023 Model Poisoning



https://owasp.org/www-project-machine-learning-security-top-10/

ML01:2023: Input Manipulation Attack

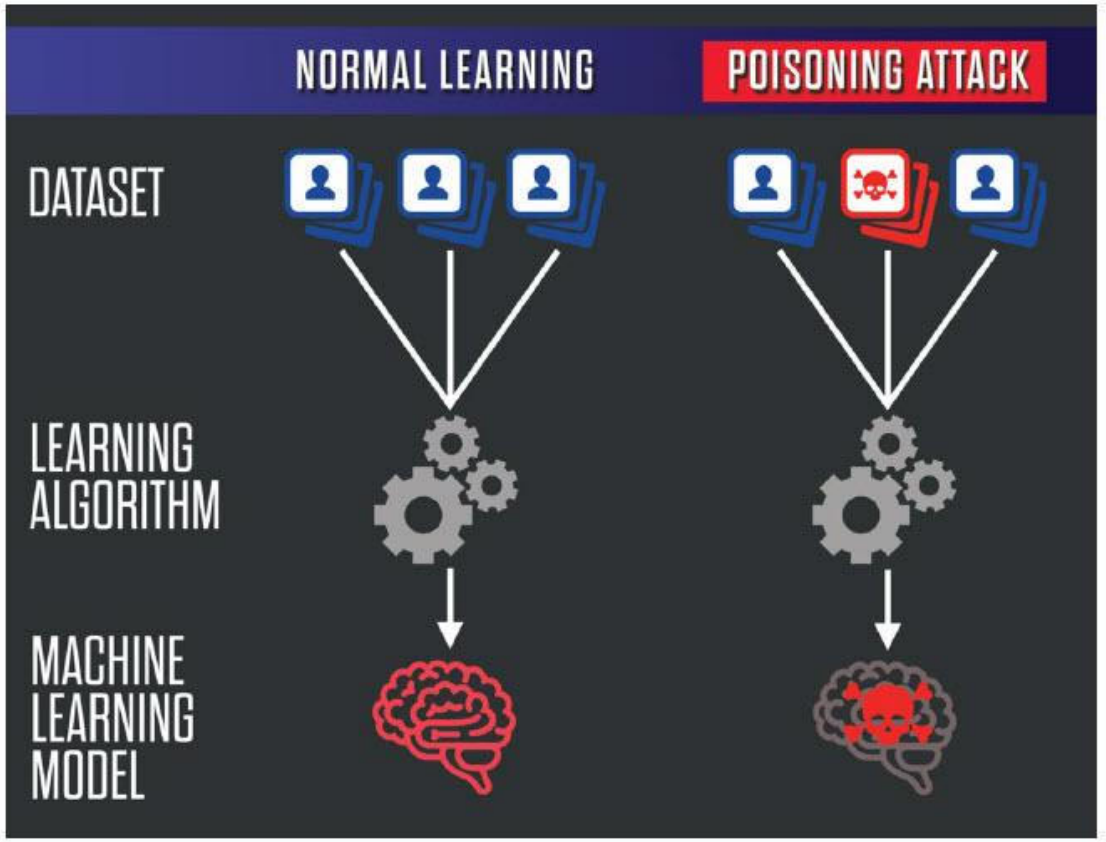
- **Description:** The attacker deliberately alters input data to mislead the model
- **Stage:** AI model use/exploitation
- **Examples:** See picture (!)
- **AI-specific mitigation:**
  - Input validation: Probably the easiest to do, false positives pose a challenge
  - Adversarial training: Attackers can circumvent but need more effort
  - Robust models: Usually perform worse in their base task (!)



Kotyan, S. (2023). A reading survey on adversarial machine learning: Adversarial attacks and their understanding. *arXiv preprint arXiv:2308.03363*.

ML02:2023: Data Poisoning Attack

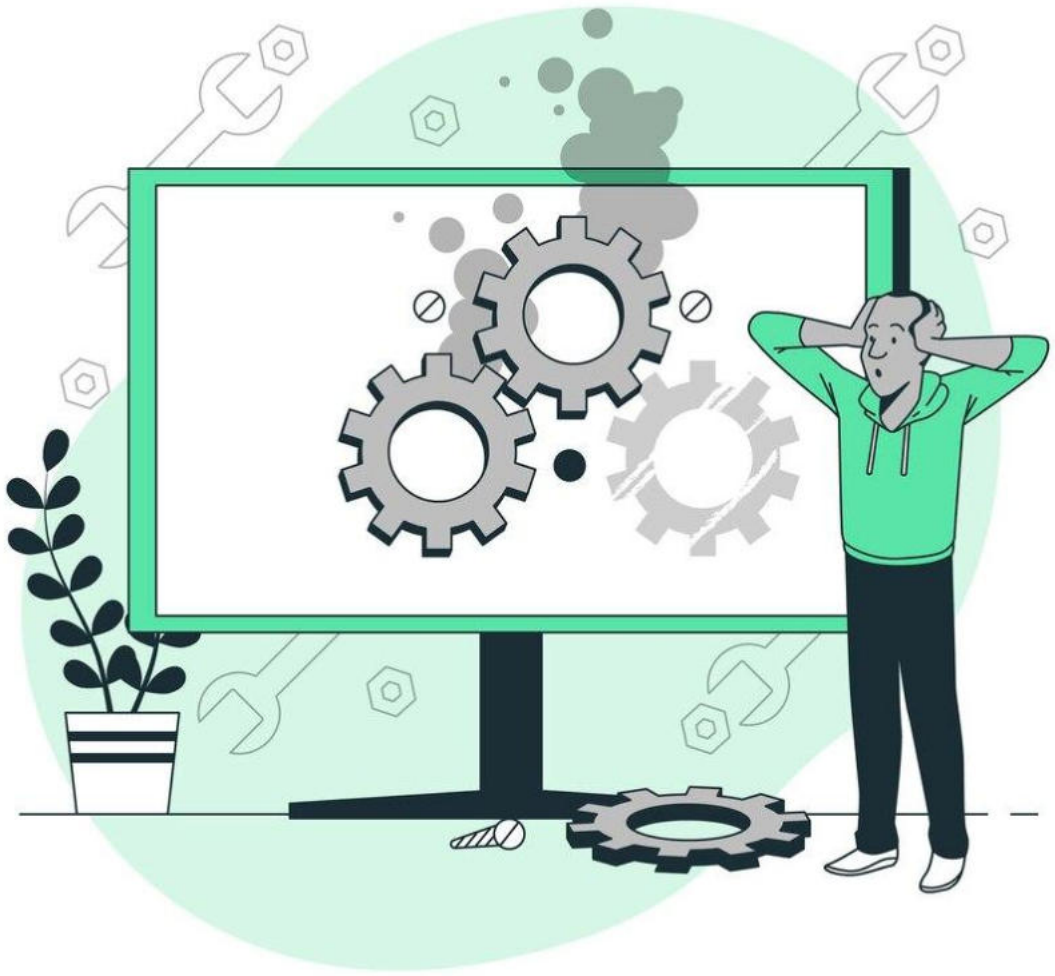
- **Description:** The attacker manipulates the training data to achieve their goals → might be a preparation of an evasion attack
- **Stage:** Training phase
- **Example(s):**
  - Compromise training data storage → Insert mis-labeled (e.g. spam) data → Mislead model (i.e., avoid detection)
- **AI-specific mitigation:**
  - Training data validation
  - Model validation & ensembles
  - Anomaly detection



https://informationmatters.net/data-poisoning-ai/

ML03:2023: Model Inversion Attack

- **Description:** The attacker observes model output with the intention to reveal sensitive information contained in training data.
  - Interact with model, observe outputs, infer training data.
  - Overfit models more susceptible. **Discuss:** Why?
- **Stage:** AI model use/exploitation
- **Example:**
  - Infer sensitive information about input data based on output observation
  - **Discuss:** Weak examples very similar to membership inference → Any additional ideas?
- **AI-specific mitigation:**
  - Input data validation → Very difficult in this scenario
  - Model retraining → Harder to 'hit' a moving target
  - Regularization
  - Explainable AI and model transparency
  - Federated learning



https://www.freepik.com/free-vector/computer-troubleshooting-concept-illustration_19184617.htm

ML04:2023: Membership Inference Attack

- **Description:** The goal is to determine if a specific observation was in the training data or not
  - AI model output (often) differ for data 'seen' compared to data 'not seen' → confidence scores are good indicators
  - Shadow models trained on similar data (!) → **Discuss:** How are these used?
- **Stage:** AI model use/exploitation
- **Examples** (nothing proven in the wild):
  - Healthcare and medical data
- **AI-specific mitigation:**
  - Differential privacy
  - Regularization

https://www.freepik.com/free-photo/flat-lay-boss-sticker-mug_9640630.htm

ML05:2023: Model Theft

- **Description:** The attacker gains unauthorized access to the trained model
  - Threat sources: API vulnerabilities, insiders
- **Stage:** AI model use/exploitation
- **Example** (nothing proven in the wild):
  - Industrial competitor steals model to gain market advantage
  - Cyber criminals steal and resell AI models
- **AI-specific mitigation:**
  - Model watermarking
  - Model obfuscation

https://www.freepik.com/free-vector/armed-burglars-committing-crimes-flat-icons-set-white-background-isolated-vector-illustration_4186278.htm

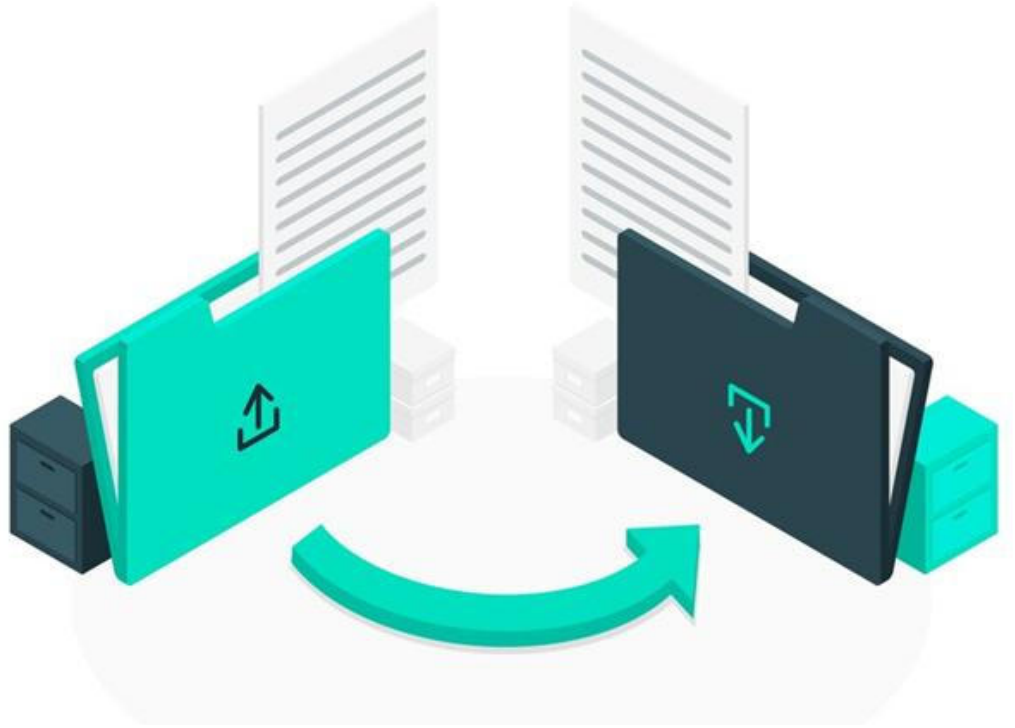
ML06:2023: AI Supply Chain Attacks

- **Description:** The attacker targets the supply chain of AI operators. Supply chain elements: MLOps, data mgmt., model mgmt., model hubs
- **Stage:** Both training and exploitation
- **Example** (nothing proven in the wild):
  - The attackers alters upstream data, models or MLOps platforms
- **AI-specific mitigation:**
  - Model IO monitoring and anomaly detection → **Discuss:** How and what to measure?

https://www.freepik.com/free-vector/flat-design-import-export-infographic_20289175.htm

ML07:2023: Transfer Learning Attack

- **Description:** A malicious actor alters a target model by influencing transfer learning.
  - Transfer learning = A model trained on one task is reused as a starting point for a model on a second task
- **Stage:** Both stages, but the desired impact is usually in exploitation
- **Example** (nothing proven in the wild):
  - The attackers alters frequently used ‘base’ model → **Discuss:** Relation to model poisoning?
- **AI-specific mitigation:**
  - Model IO monitoring and anomaly detection → **Discuss:** How and what to measure?



https://www.freepik.com/free-vector/file-transfer-concept-illustration_5843118.htm

ML08:2023: Model Skewing

- **Description:** The attacker modifies training data distribution → Model behaves not as planned.
  - **Discuss:** Relation to data poisoning?
- **Stage:** Training
- **Example** (nothing proven in the wild):
  - **Discuss:** Credit rating, criminal rating (?)
- **AI-specific mitigation:**
  - Model IO and performance monitoring, as well as anomaly detection
  - Model re-training
  - Model robustness testing → Testing model sensitivity to select training data changes

https://www.freepik.com/free-vector/file-transfer-concept-illustration_5843118.htm

ML09:2023: Output Integrity Attack

- **Description:** The attacker manipulates the output of an AI system to reach their goals.
- **Stage:** AI model use/exploitation
- **Example** (nothing proven in the wild):
  - Modify output of disease diagnosis AI → Wrong prescription with dire consequences
- **AI-specific mitigation:**
  - Model IO and performance monitoring, as well as anomaly detection

https://www.freepik.com/free-vector/file-transfer-concept-illustration_5843118.htm

ML10:2023: Model Poisoning

- **Description:** The attacker obtain access to and modifies the AI model parameters.
- **Stage:** AI model use/exploitation
- **Example** (nothing proven in the wild):
  - Credit risk, disease diagnosis, anomaly detection (as in spam and IDS)
- **AI-specific mitigation:**
  - AI system monitoring
  - Model regularization
  - Use of robust models
  - Use of model ensembles → **Discuss:** Which are the edge cases in which this helps?

https://www.freepik.com/free-vector/file-transfer-concept-illustration_5843118.htm

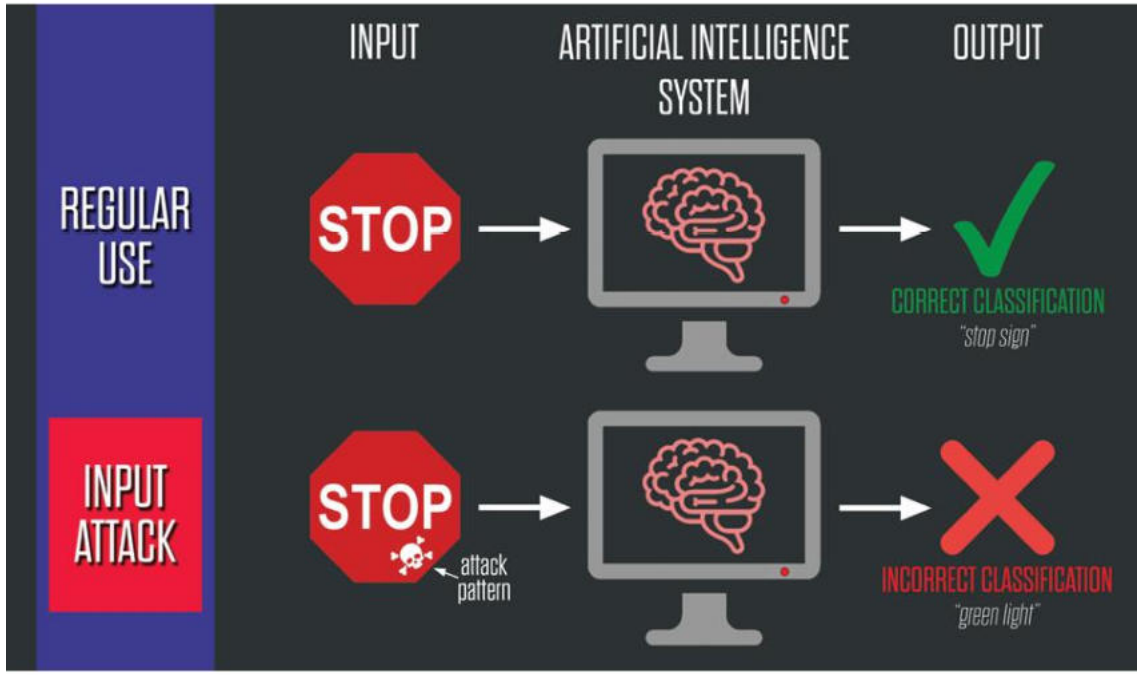
Training security & OWASP Top 10

- (Training) data poisoning → The attacker obtains access to training data and modifies as suitable for him/her (ML02:2023)
  - Examples: email spam, network security monitoring (NSM) analysis
- AI Supply Chain Attacks (ML06:2023) → Attack surface includes MLOps platforms; data/model management platforms (Kaggle) & model hubs (Huggingface); other specialized software
  - NOTE: Relevant in both the training & exploitation stages
- Transfer learning attack → The attacker trains a model on one task and then maliciously fine tunes it on another task (ML07:2023)
- Model Skewing → The attacker manipulates the distribution of the training data → The model behaves in an undesirable way (ML08:2023)

https://owasp.org/www-project-machine-learning-security-top-10/

AI in use & OWASP Top 10

Data Acquisition | Data Pre-processing | Training | Exploitation



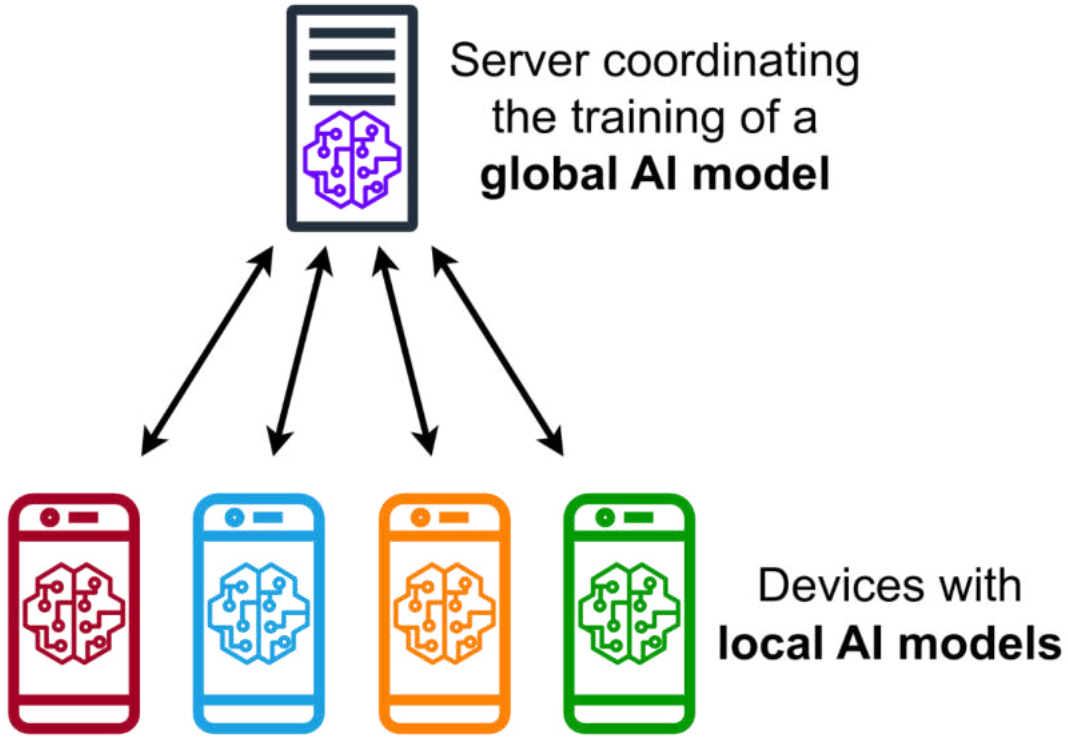
https://www.belfercenter.org/publication/AttackingAI

- Input Manipulation Attack (ML01:2023) → See picture on left
- Model Inversion (ML03:2023) → Reverse engineer model to extract sensitive info
- Membership Inference Attack (ML04:2023)
- Model Theft (ML05:2023)
- Output Integrity Attack (ML09:2023)
- Model Poisoning (ML10:2023)

https://owasp.org/www-project-machine-learning-security-top-10/

Data Security challenges

- Data security is of high relevance in both stages and most OWASP Top 10 vulnerabilities
- Main challenges:
  - Data poisoning → Attacker modifies the data used in training
  - Data leak I → Training data theft
  - Data leak II → Sensitive data access in the exploitation stage (e.g., via inversion)
- Possible solutions:
  - Anonymization and differential privacy
  - Synthetic data
  - Federated learning



https://en.wikipedia.org/wiki/Federated_learning

OWASP Top 10 ML Overview

Title	Training	Use	AI-specific mitigations
ML01:2023 Input Manipulation Attack		X	Robust models, adversarial training, input validation
ML02:2023 Data Poisoning Attack	X	(X)	Training data validation, model validation, model ensembles, monitoring & anomaly detection
ML03:2023 Model Inversion Attack		X	Input data validation, retraining, explainable AI/model transparency, federated learning
ML04:2023 Membership Inference		X	Differential privacy, regularization
ML05:2023 Model Theft		X	Watermarking, obfuscation
ML06:2023 AI Supply Chain Attacks	X	(X)	monitoring and anomaly detection
ML07:2023 Transfer Learning Attack	X	(X)	monitoring and anomaly detection
ML08:2023 Model Skewing	X		monitoring and anomaly detection, re-training, robustness testing
ML09:2023 Output Integrity Attack		X	monitoring and anomaly detection
ML10:2023 Model Poisoning	(X)	X	Regularization, robust models, re-training

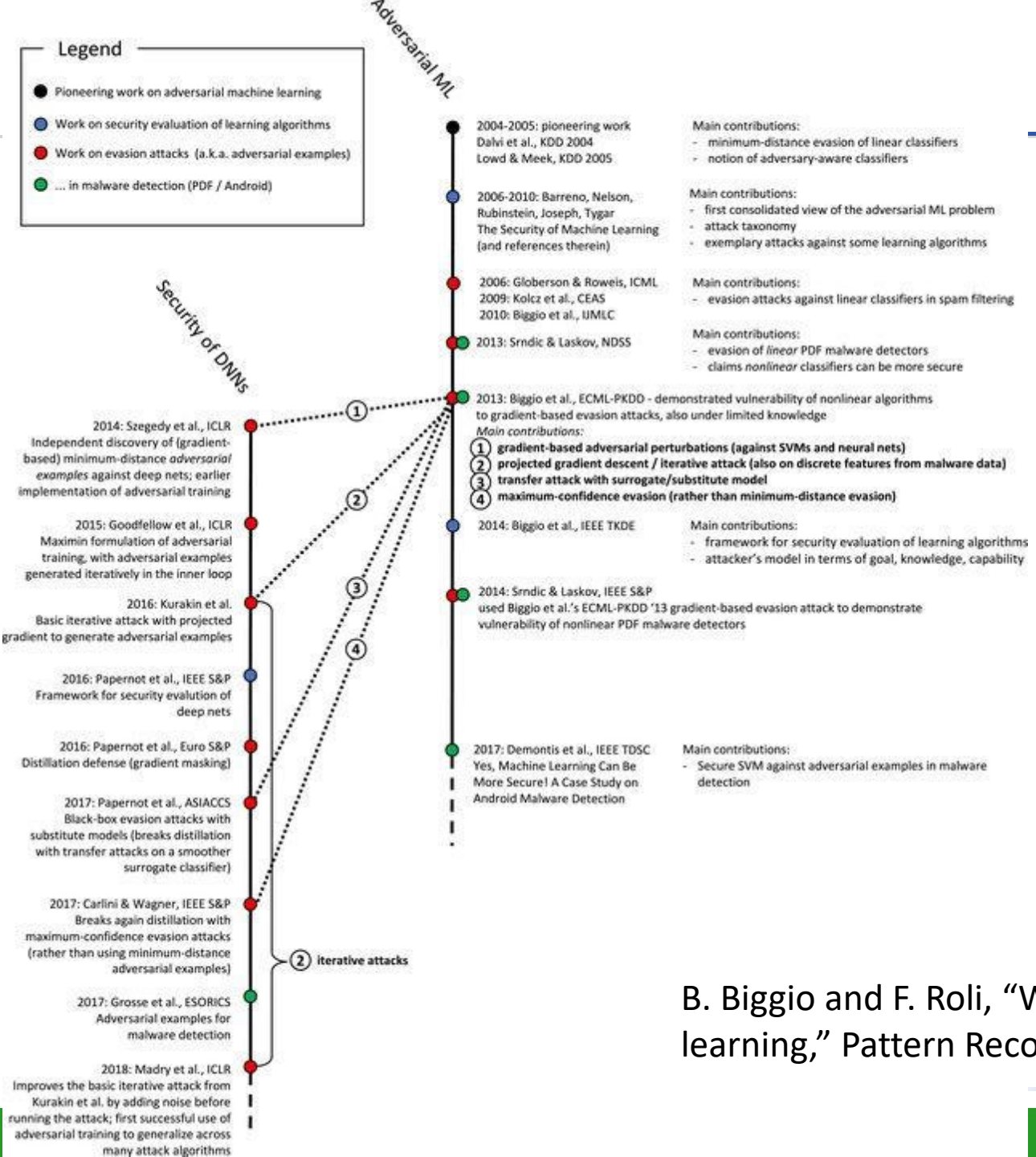
OWASP Top 10 for Large Language Model Applications

- LLM01:2025 Prompt Injection
- LLM02:2025 Sensitive Information Disclosure
- LLM03:2025 Supply Chain
- LLM04: Data and Model Poisoning
- LLM05:2025 Improper Output Handling
- LLM06:2025 Excessive Agency
- LLM07:2025 System Prompt Leakage
- LLM08:2025 Vector and Embedding Weaknesses
- LLM09:2025 Misinformation
- LLM10:2025 Unbounded Consumption

https://genai.owasp.org/resource/owasp-top-10-for-llm-applications-2025/

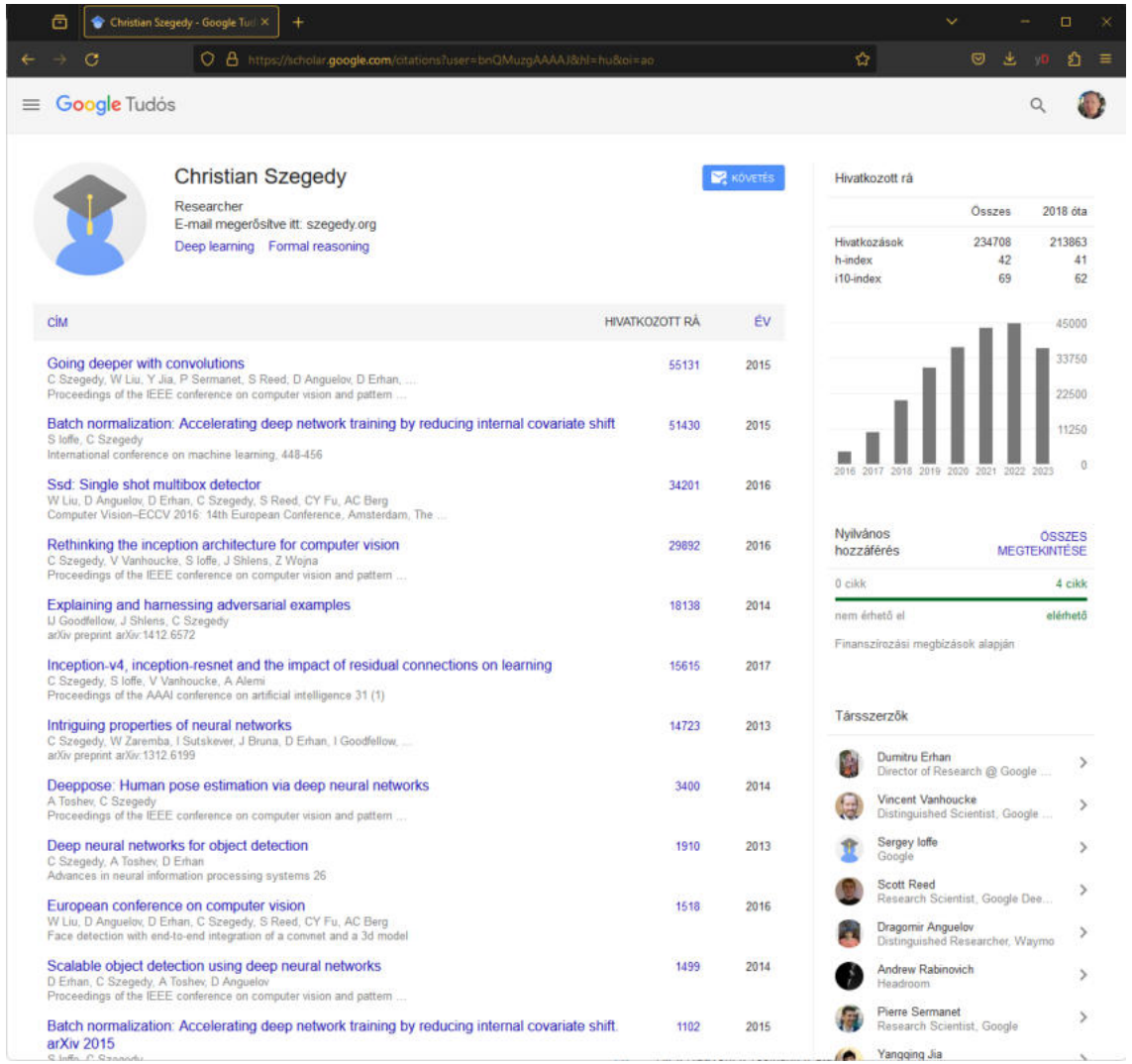
WHO IS WHO IN AI SECURITY?

Adversarial learning timeline (2004-2018)



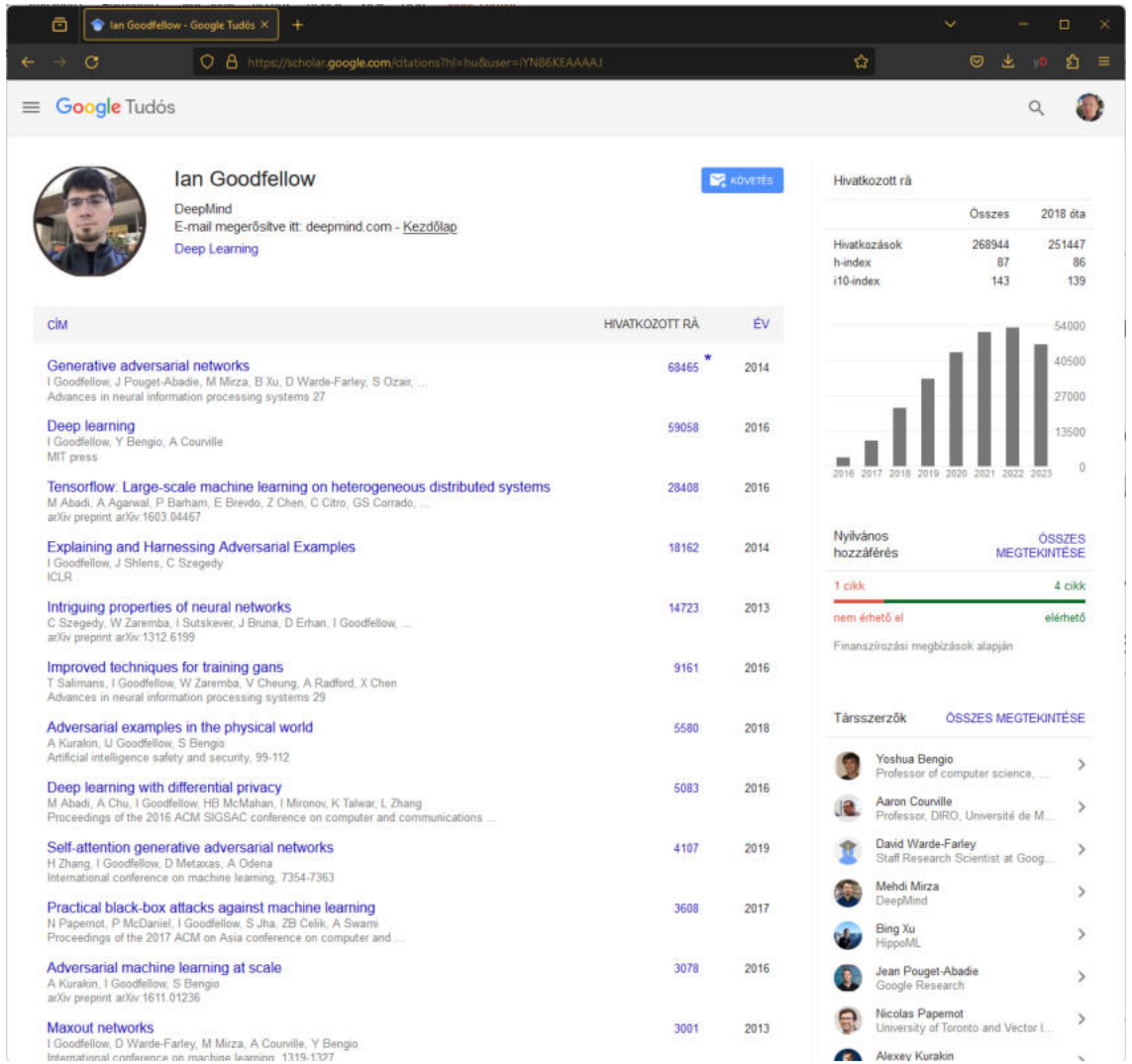
B. Biggio and F. Roli, "Wild patterns: Ten years after the rise of adversarial machine learning," Pattern Recognition, vol. 84, pp. 317–331, 2018.

Christian Szegedy (Google)



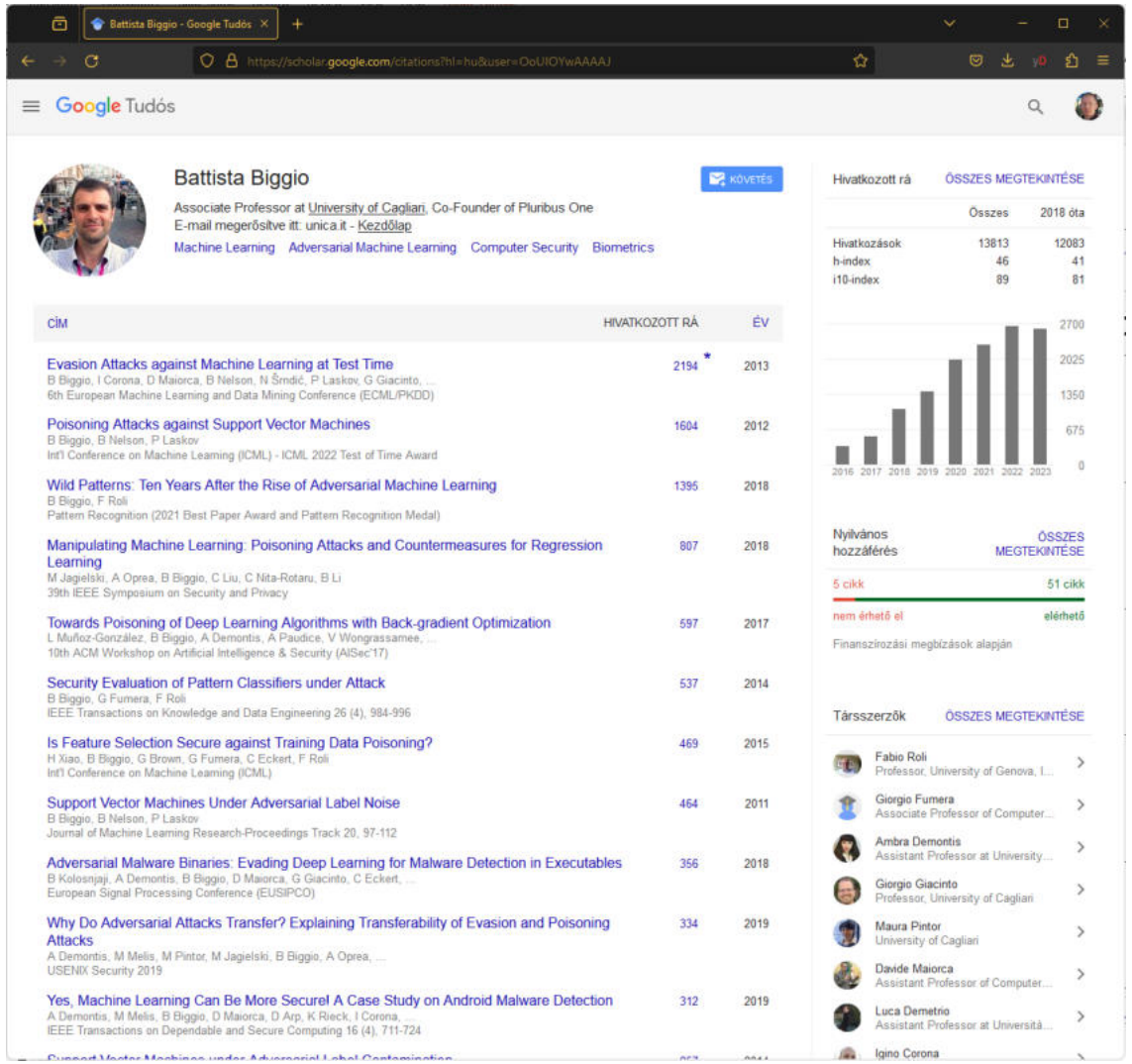
- **Workplace:** Google
- **Topics:**
  - Convolutional networks
  - Object identification in pictures
  - Adversarial samples in image analysis
- **Google Scholar citations:** https://scholar.google.com/citations?user=bnQMuzgAAAAJ&hl=hu&oi=ao

Ian Goodfellow



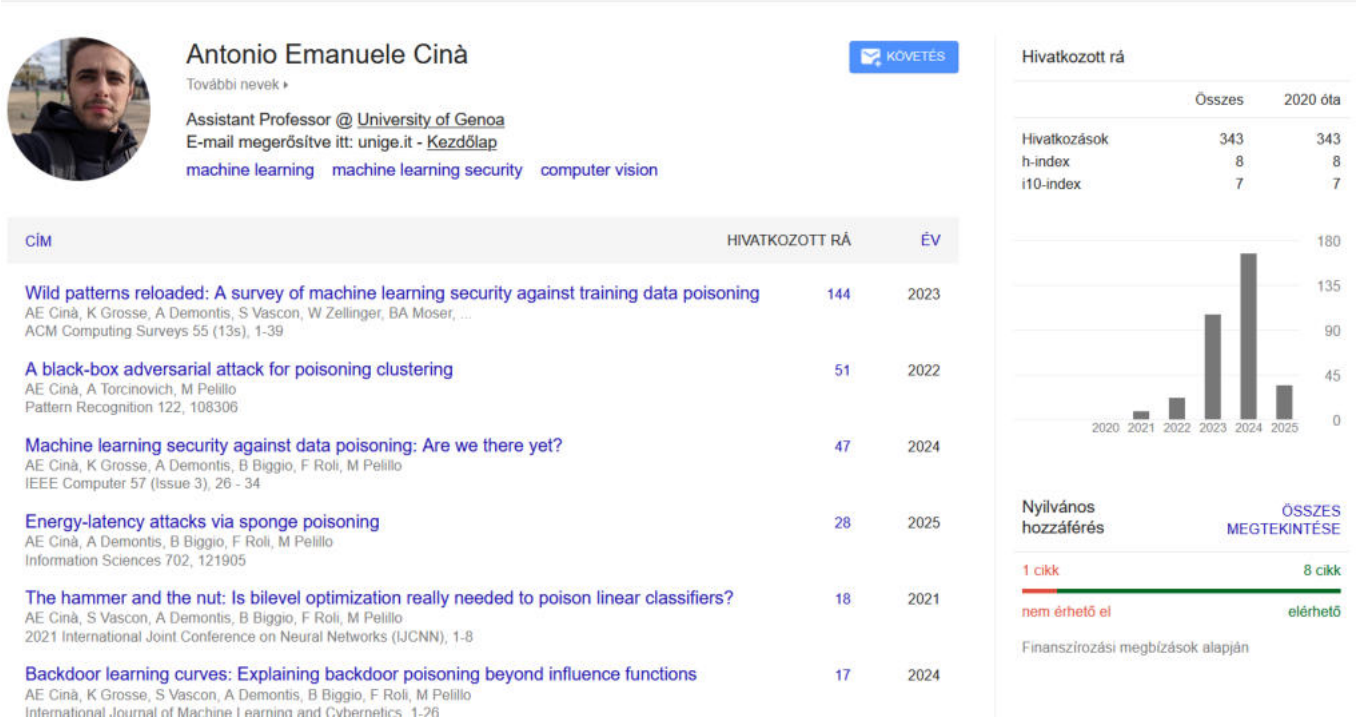
- **Workplace:** DeepMind (Google)
- **Top scientific interests :**
  - Deep neural network analysis
  - GAN
  - Adversarial samples in machine learning
- **Google Scholar citations:** https://scholar.google.com/citations?hl=hu&user=iYN86KEAAAAJ

Battista Biggio (Uni Cagliari)



- **Workplace:** University of Cagliari, Italy
- **Top scientific interests:**
  - Modell poisoning
  - Evasion attacks
  - Regression attacks
  - Adversarial software analysis
- **Google Scholar citations:** https://scholar.google.com/citations?hl=hu&user=OoUIOYwAAAAJ

Antonio Emanuele Cinà



Antonio Emanuele Cinà

További nevek ▸

KÖVETÉS

Assistant Professor @ University of Genoa
E-mail megerősítve itt: unige.it - Kezdőlap
machine learning machine learning security computer vision

CÍM	HIVATKOZOTT RÁ	ÉV
Wild patterns reloaded: A survey of machine learning security against training data poisoning AE Cinà, K Grosse, A Demontis, S Vascon, W Zellinger, BA Moser, ... ACM Computing Surveys 55 (13s), 1-39	144	2023
A black-box adversarial attack for poisoning clustering AE Cinà, A Torcinovich, M Pelillo Pattern Recognition 122, 108306	51	2022
Machine learning security against data poisoning: Are we there yet? AE Cinà, K Grosse, A Demontis, B Biggio, F Roli, M Pelillo IEEE Computer 57 (Issue 3), 26 - 34	47	2024
Energy-latency attacks via sponge poisoning AE Cinà, A Demontis, B Biggio, F Roli, M Pelillo Information Sciences 702, 121905	28	2025
The hammer and the nut: Is bilevel optimization really needed to poison linear classifiers? AE Cinà, S Vascon, A Demontis, B Biggio, F Roli, M Pelillo 2021 International Joint Conference on Neural Networks (IJCNN), 1-8	18	2021
Backdoor learning curves: Explaining backdoor poisoning beyond influence functions AE Cinà, K Grosse, S Vascon, A Demontis, B Biggio, F Roli, M Pelillo International Journal of Machine Learning and Cybernetics, 1-26	17	2024

Hivatkozott rá

	Összes	2020 óta
Hivatkozások	343	343
h-index	8	8
i10-index	7	7

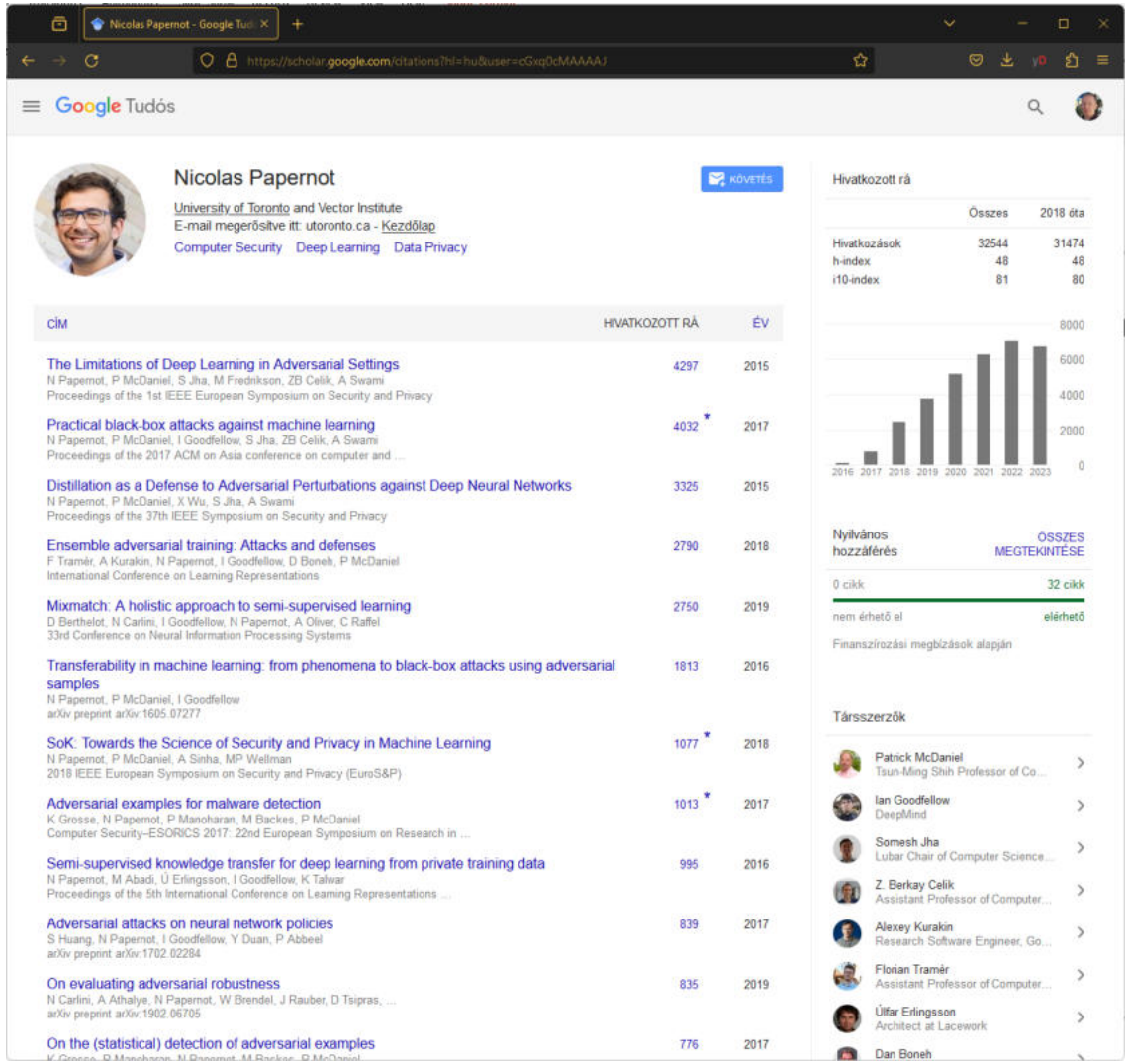
Nyilvános hozzáférés ÖSSZES MEGTEKINTÉSE

1 cikk 8 cikk
nem érhető el elérhető

Finanszírozási megbízások alapján

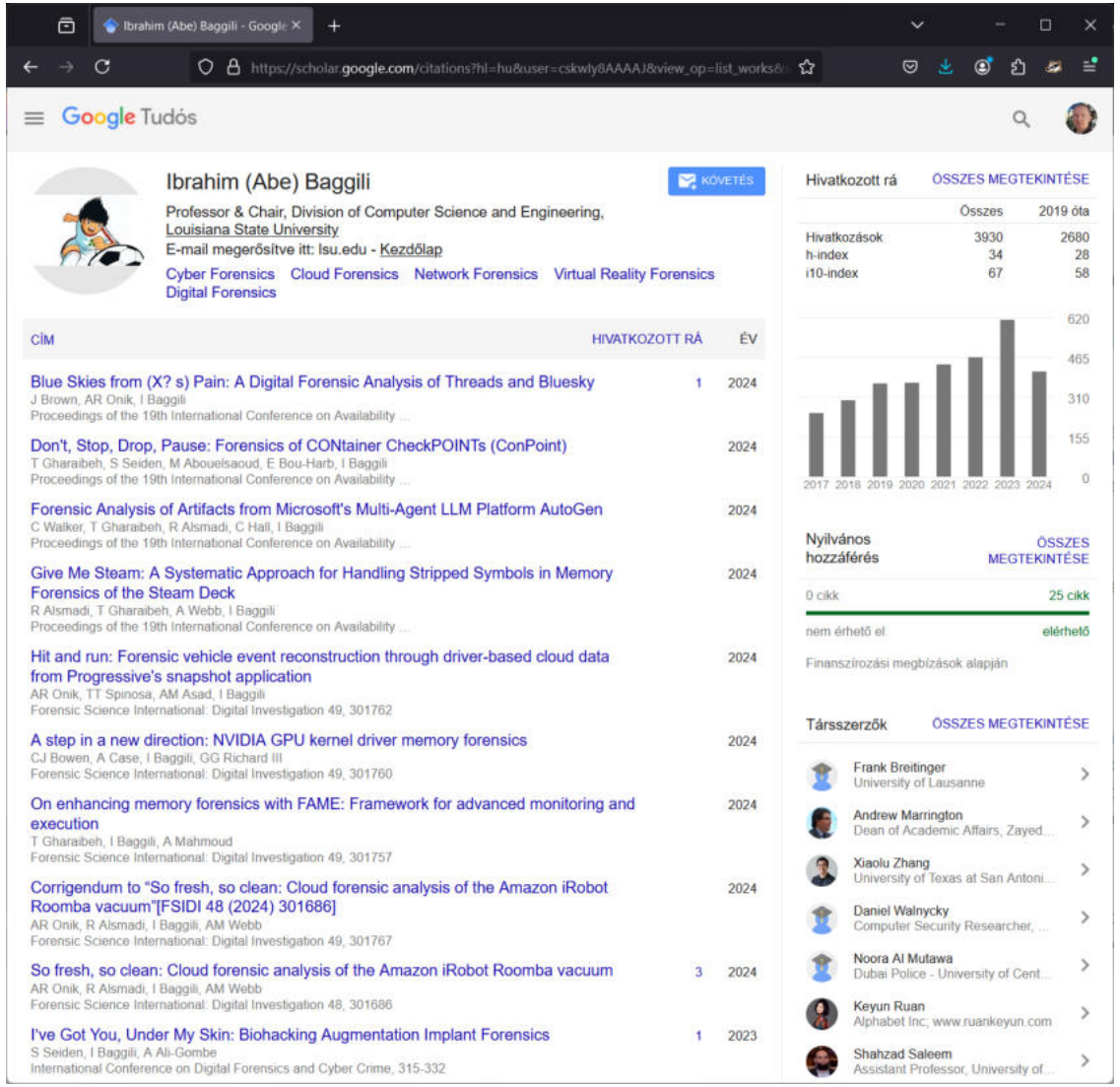
- **Workplace:** University of Genoa, Italy
- **Top scientific interests:**
  - Machine Learning Security and Reliability
  - Cybersecurity and AI for Scam Detection
- **Google Scholar citations:** https://scholar.google.com/citations?user=Qtj8Lb8AAAAJ&hl=hu&oi=ao
- **Github:** https://cinofix.github.io/

Nicolas Papernot



- **Workplace:** University of Toronto, Canada
- **Top scientific interests:**
  - Deep learning in adversarial environments
  - Blackbox attacks
- **Google Scholar citations:** https://scholar.google.com/citations?hl=hu&user=cGxq0cMAAAAJ

Ibrahim (Abe) Baggili



- **Workplace:** Louisiana State University, USA
- **Top scientific interests:**
  - AI forensics (plenary at ARES 2023)
  - Digital forensics background
- **Google Scholar citations:** https://scholar.google.com/citations?hl=hu&user=cskwly8AAAAJ&view_op=list_works&sortby=pubdate

Summary

- OWASP Top 10 ML
- AI security timeline and ‘Who is who?’

Thank you for your attention!

Eötvös Loránd University (ELTE)
Faculty of Informatics (IK)
Pázmány Péter sétány 1/c
1117 Budapest, Hungary

INPUT DATA MANIPULATION

Lecture #5 in Introduction to Data Security

Imre Lendák, PhD, GICSP

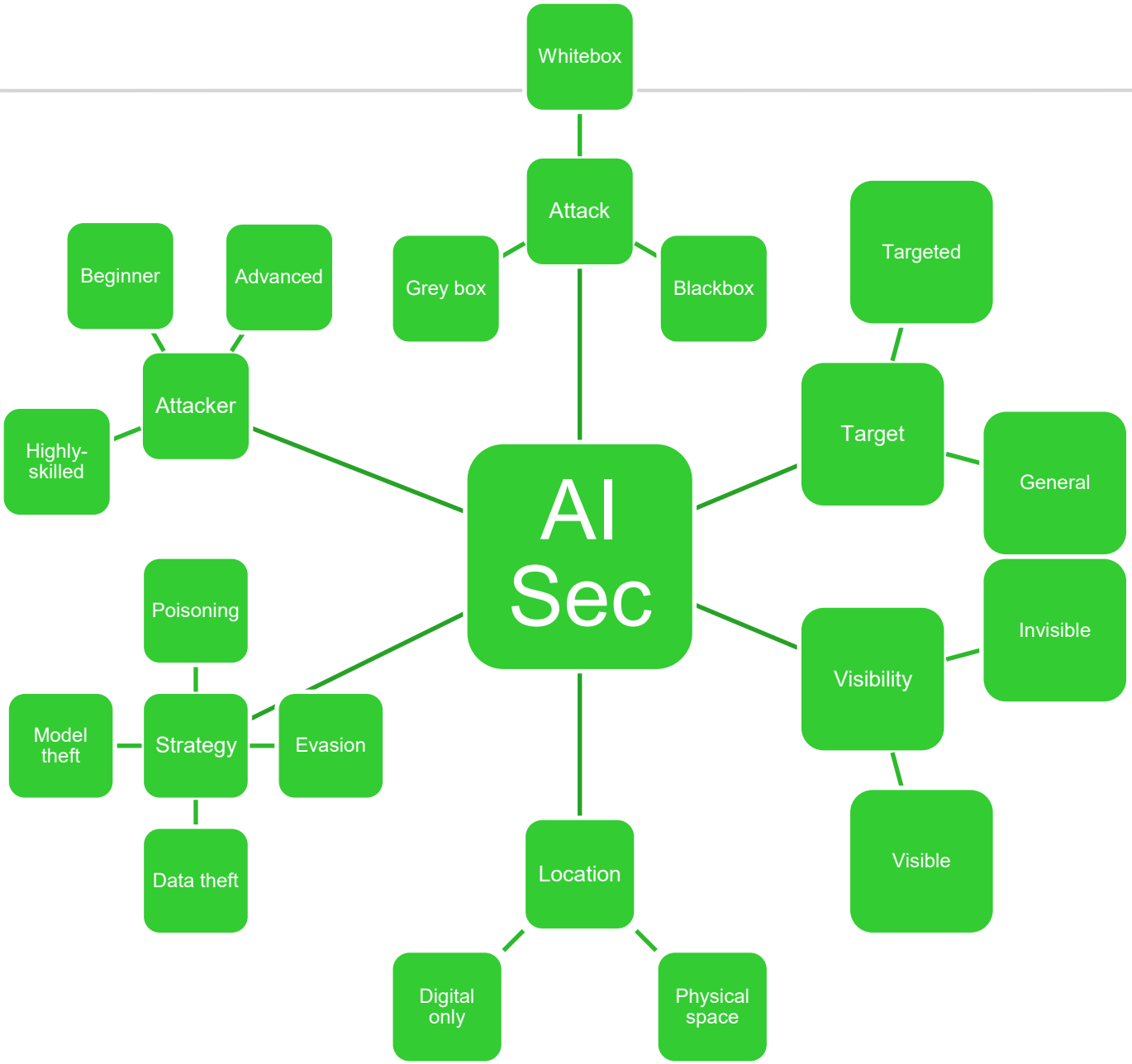
2026
Budapest, Hungary

Presentation outline

- Input data manipulation in ML tasks
  - Link: OWASP/ML01:2023: Input Manipulation Attack
  - Note: Not just evasion (!)
  - Input manipulation vs data type
    - Prompt injection attacks against LLMs

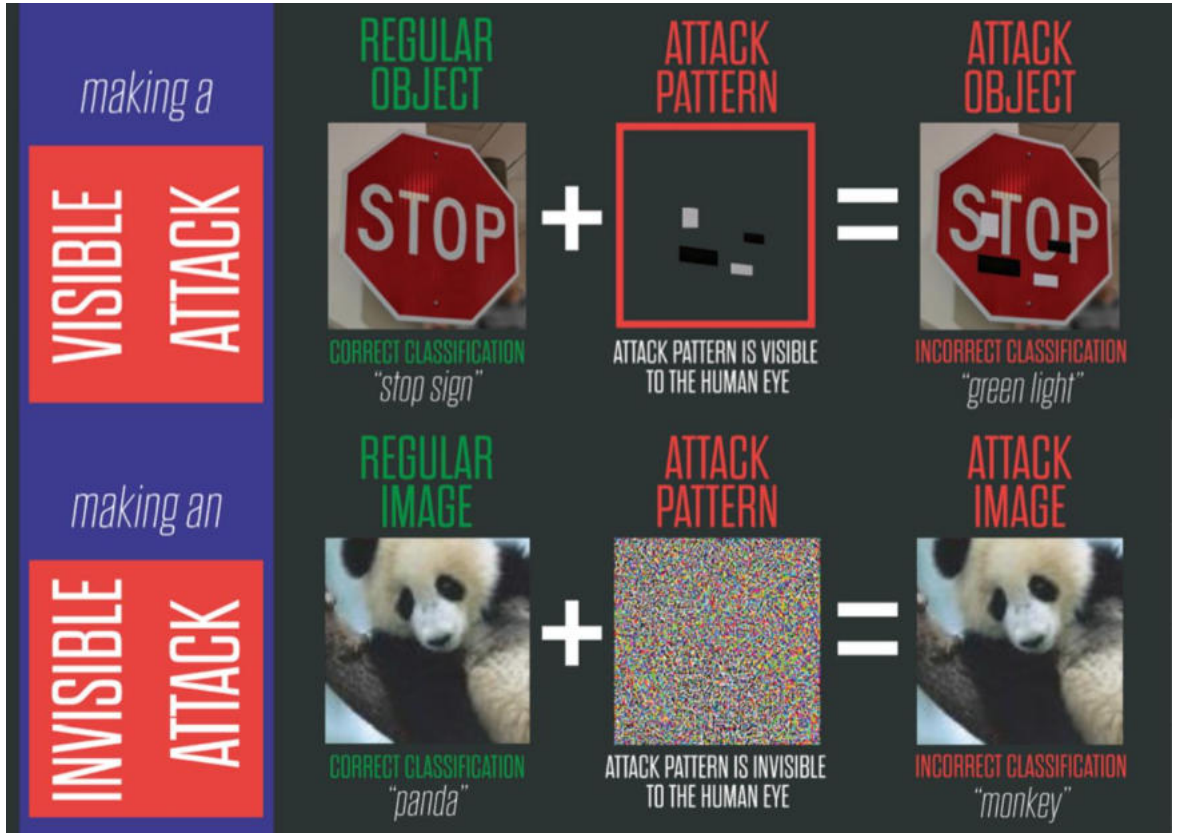
- Black box attacks
- White box attacks

Input data manipulation in applied AI



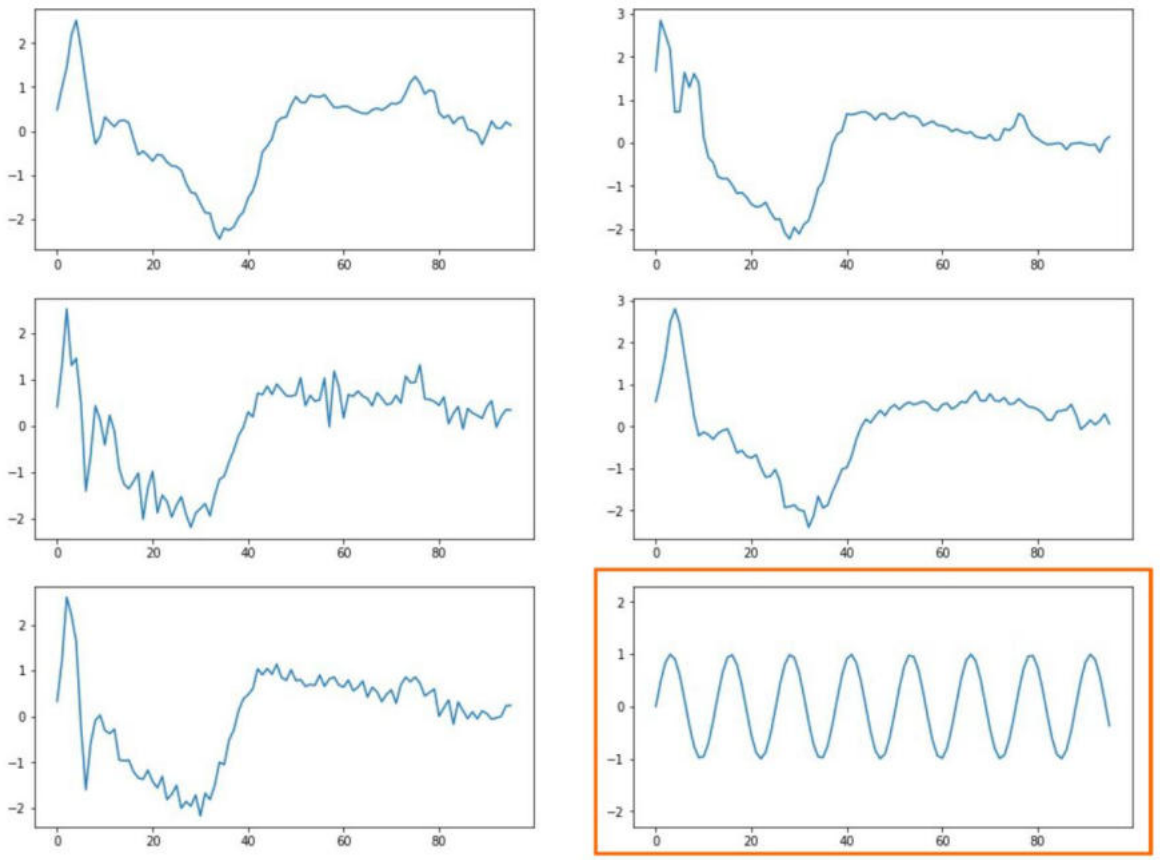
Classification

- Attacks against binary classification:
  - Avoid spam detection
  - Malware detection → Manipulate (known) malware samples to evade detection
  - **Note:** Above examples can be approached with text mining
- Attacks against applied multi-class classification:
  - Avoid face recognition → Isolate face, compare against database, assess similarity
  - Mislead smart vehicle 'vision': recognition of known objects on the road and on the side of the road



Anomaly and change detection

- Intrusion detection systems
  - Hide malicious activity in otherwise legit protocols or activities → Needle in the haystack problem
- Anomaly and change detection (both!) in endpoint (equipment or employee) behavior
  - Warm up and slightly changed behavior
- **Discuss:** Anything else?

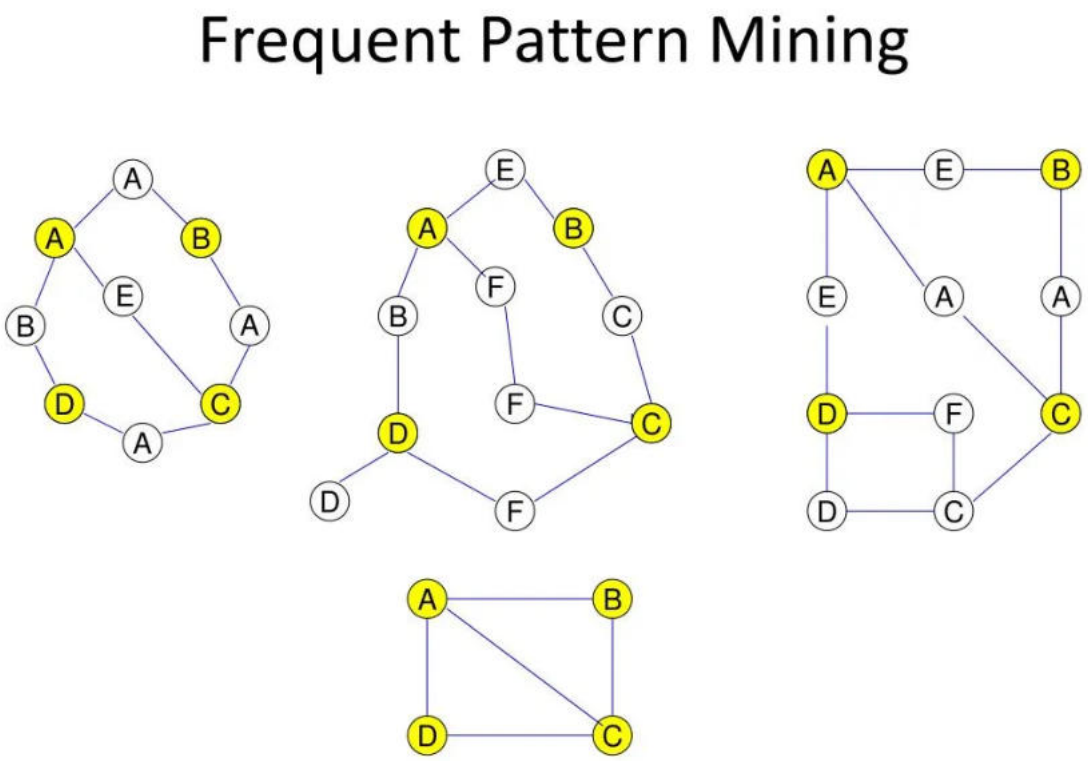


Buza K. Time Series Classification and its Applications, 8th International Conference on Web Intelligence, Mining and Semantics. June 25 – 27 2018, Novi Sad, Serbia.

Clustering

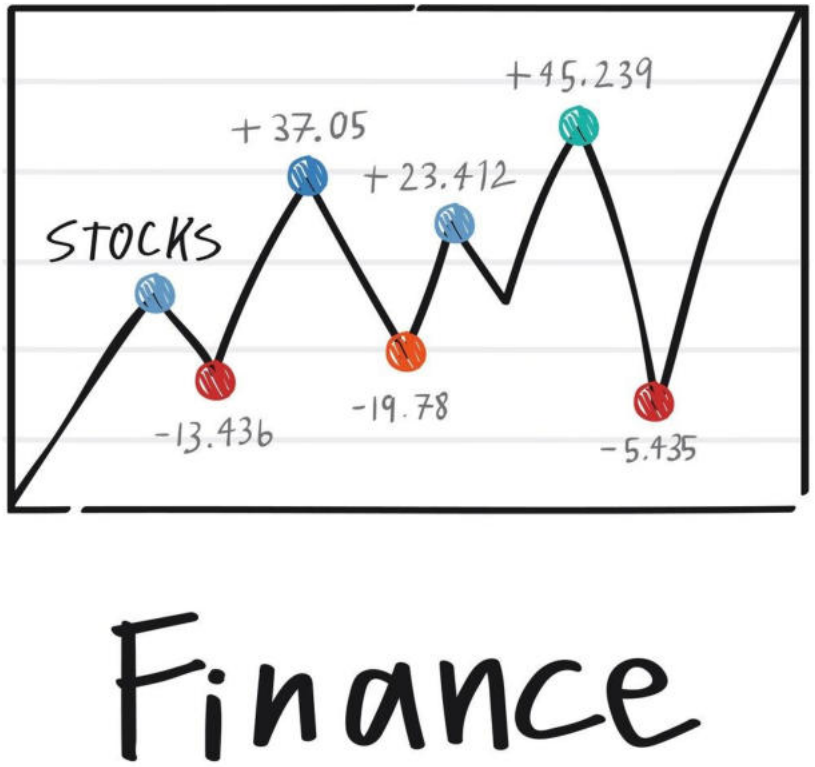
- Clustering algorithms ‘group’ similar data points/observations
- Possible attacker goals:
  - Craft malicious input data for systems in which clusters are used in downstream classification tasks → Unsupervised anomaly detection
  - Submit torrent of unwanted data points → Alter clusters in re-training
  - **Discuss:** Anything else?

Frequent pattern mining



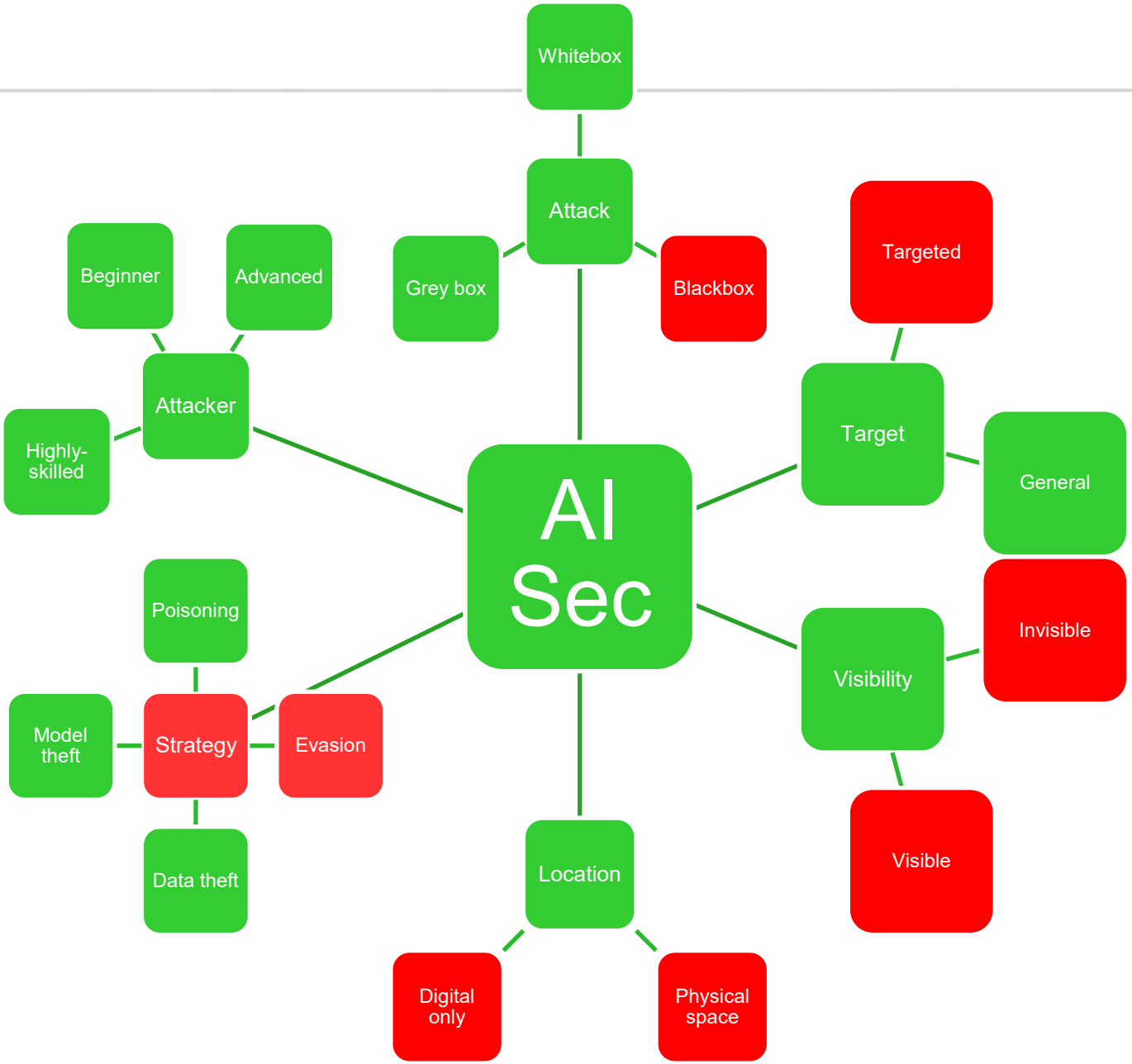
- Frequent pattern mining algorithm identify data occurring jointly or in sequence
- Possible attacks:
  - Bots generate sequences of online actions → Trick recommender engines into suggesting certain products
- **Discuss:** What else?

Regression



- Regression algorithms fit shapes to known observations and allow prediction(s)
- **Discuss:** Possible attacks against regression in use?

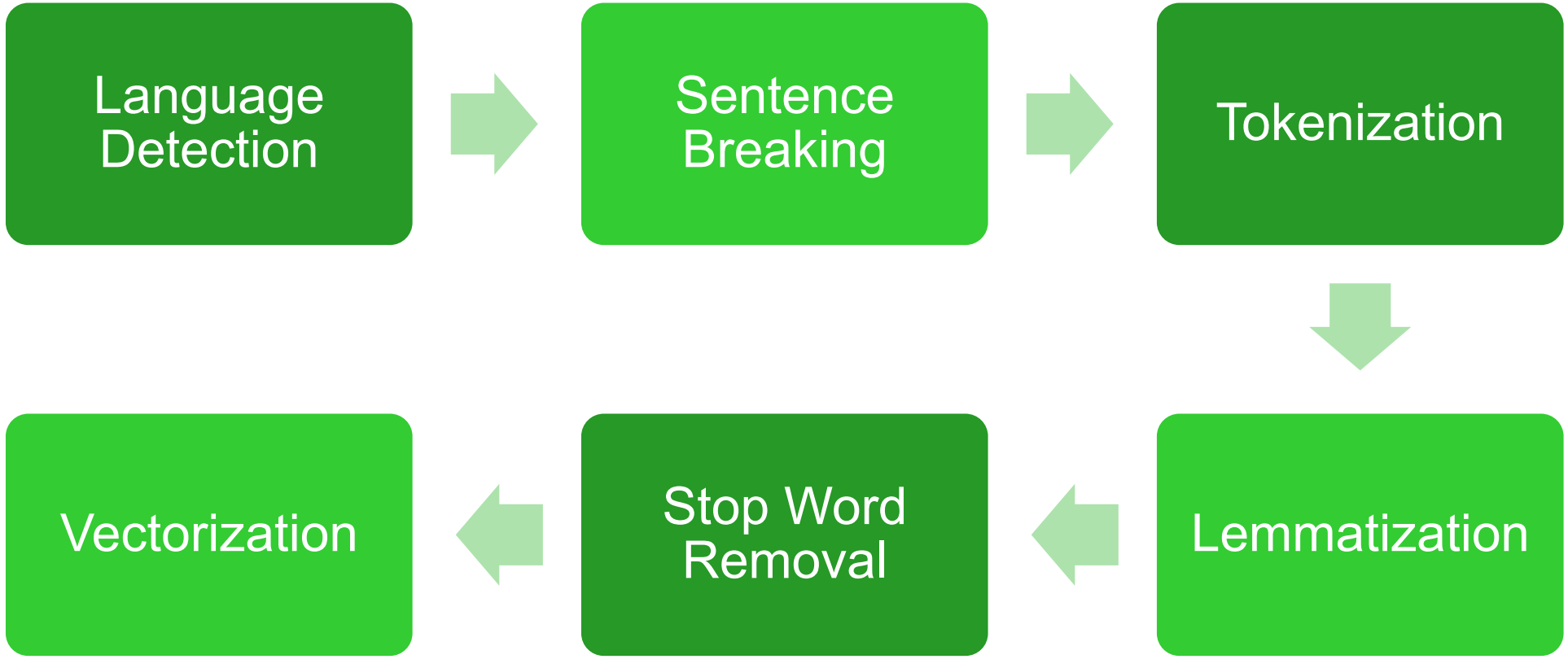
ATTACK PATTERNS IN TEXTUAL DATA



Attack patterns in text analysis

- **Legitimate text analysis tasks:**
  - Scan to electronic text
  - Text comprehension
  - Text classification
  - Text generation
  - (Less frequently) Authentication based on typing
- **ML model:** deep learning, different classification models
- **Possible attacker goals:**
  - Evade spam/malware detection
  - Tamper with typing-based authentication
- **Visibility:**
  - Visibly altered print or electronic document
  - Invisible noise added to print or electronic document
- **Location:**
  - Physical space: Modify paper-based document before scanning
  - Electronic space: Scanned or digital-only document

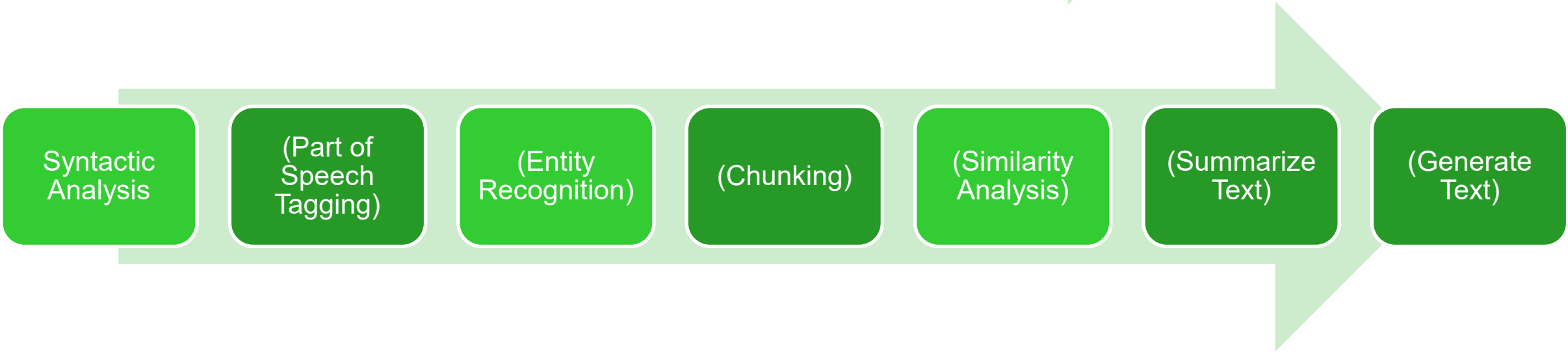
Text pre-processing steps



Discuss: Which step to attack (in general)? And how?

https://medium.com/@akankshamenon27/text-analytics-a-brief-introduction-and-steps-involved-68e882b10f34

Text analysis steps



Discuss: Which step to attack (in general)? And how?

Attack patterns in spam detection

- Initial spam detection based on keywords (e.g., Viagra, prize, Nigerian prince), server IP address or domain names:
  - Blacklisting of domains, email addresses, servers (IP address) → Rule-based AI
- Latest unwanted mail is often phishing or spear phishing
  - Tailored, evolving & multi-lingual text generated with (malicious) chatbots
  - Legitimate servers or endpoints hacked and used to generate spam

Fig. 2. Examples of *clean* (top) and *obfuscated* (bottom) spam images [57].

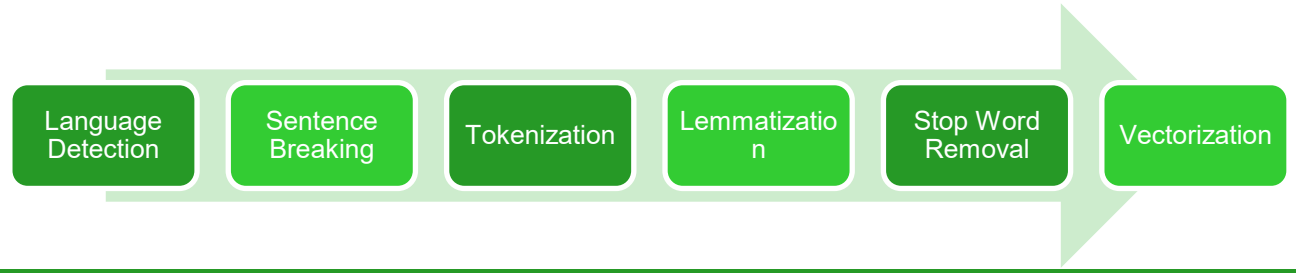
Spam arms race

Content obfuscation techniques

- Word & character manipulation
  - Misspelt words e.g., v1agra
  - Unicode chars or homoglyphs (chars which look similar)
- Image-based spam → Text is embedded in images
- HTML and CSS manipulation → Hidden text picked up by spam scanners only, text salting
- From and domain spoofing
- Rotate & 'warm up' IP addresses
- Use of compromised email accounts
- Use of chatbots to generate convincing and multi-lingual content → Discuss: Can similarity analysis be of use in this context?

ML algorithms in spam detection

- Naïve Bayes → Text analysis-based probability analysis, efficient
- Support Vector Machines (SVM) → Able to handle high-dimensional data
- Decision Trees and Random Forests → Tree-like structures to classify emails based on features
- Logistic Regression → Model probability of emails being spam using logistic function.
- (Deep) Neural Networks → Long Short-Term Networks (LSTM) and text-focused networks



Latest advances: Google's RETVec

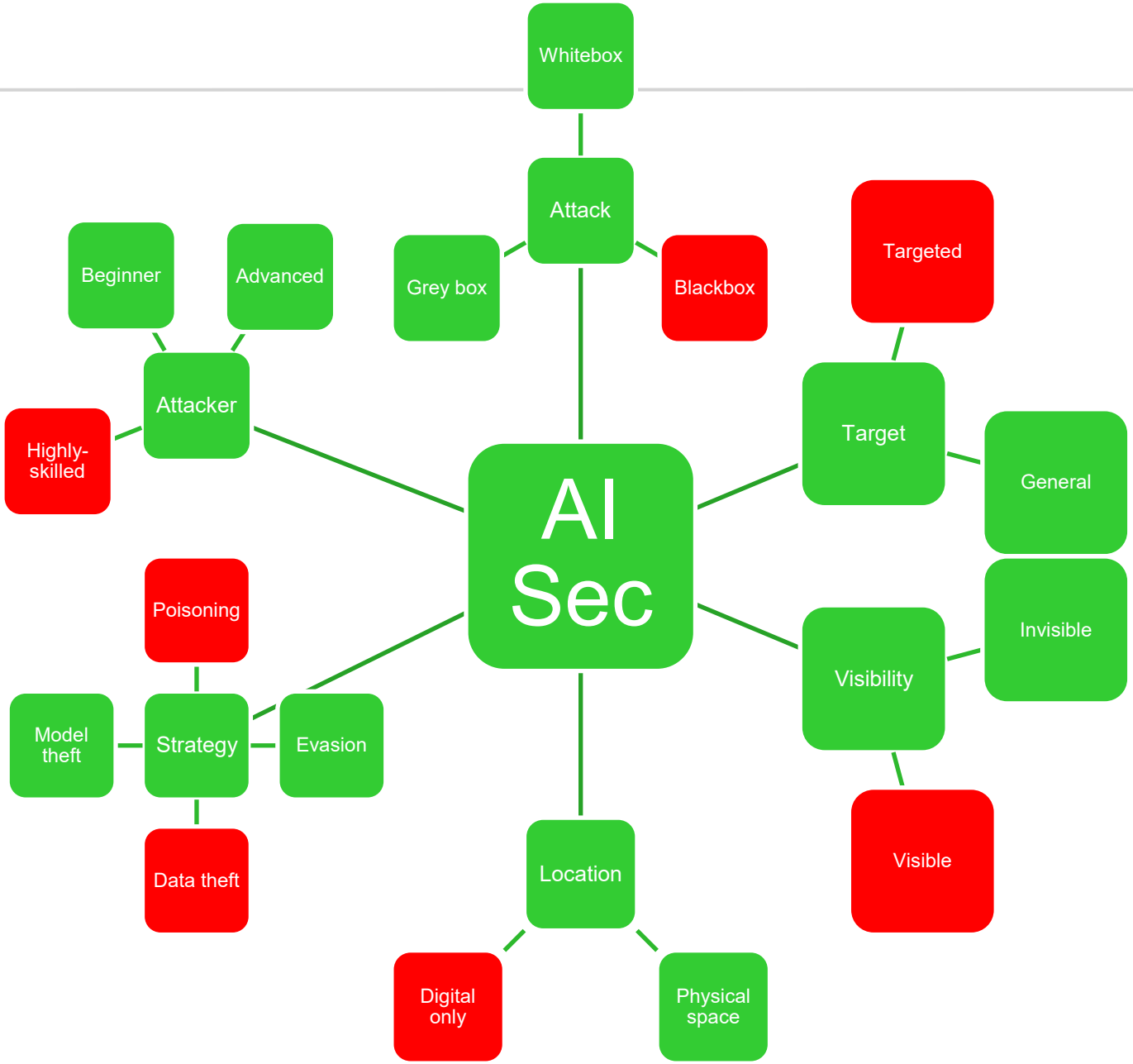
- Resilient and Efficient Text Vectorizer (RETVec) → Advanced, multi-language text vectorization → Converts text into vectors which can be analyzed by ML models
  - Supports over 100 languages
  - Similarity learning
- Resilience against:
  - Homoglyphs (similar characters)
  - Intentional typos
  - Deletion or insertion of characters
  - LEET speak e.g., T3nn1sP!4y3r

Attack patterns in malware detection

- Traditional malware analysis based on signature databases
  - Obtain new malware sample → create hash value → put hash in malware database → ship database to anti-malware users
- Attacker counter-measures
  - Fileless, ephemeral malware → No more viruses (!) → **Discuss:** Why?
  - Continuous malware code modification

- Defensive measures:
  - Disassembly of malware binaries → Code & behavior analysis
  - Identify similar behavior
  - Anomalous process behavior

ATTACK PATTERNS IN LLMS



Attack patterns in large language models

https://www.wisecube.ai/blog/a-comprehensive-overview-of-large-language-models/

- **DEF:** Latest chatbots are large language models (LLM) trained on large volumes of text which feature
  - Input data analysis & comprehension
  - Text generation as responses to questions asked by customers
  - Conversation mode → Linking multiple successive questions and answers
- Input data manipulation goals:
  - Prompt injection → Inputs which cause the LLM to generate malicious output or other unintended action
    - Mislead LLM into divulging training data
  - Sponge attack → Increase energy consumption & model overload
  - **Discuss:** Anything else?

Prompt injection attacks elaborated 1/2

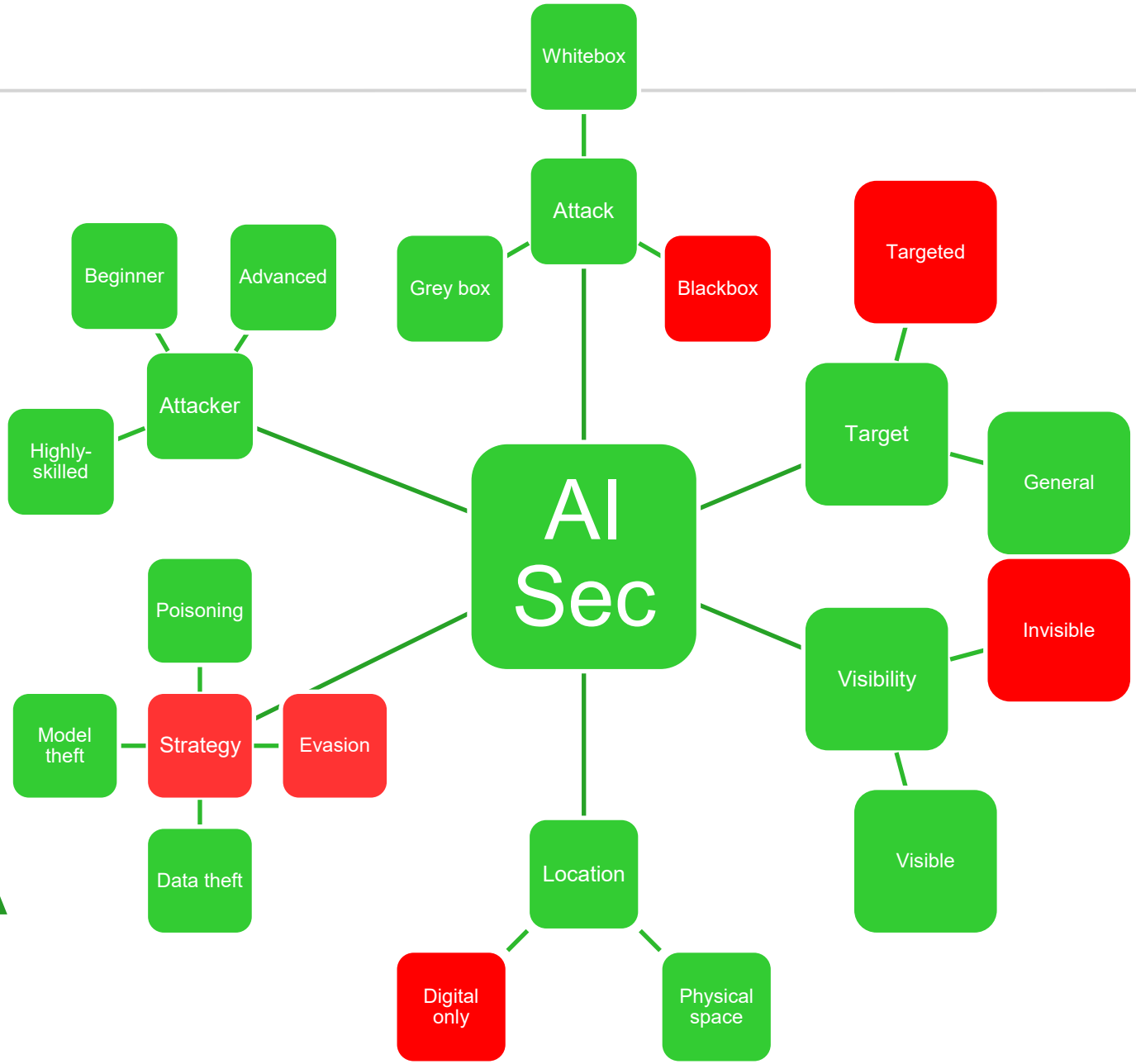
- **DEF:** In general, prompts consist of the following elements:
  - Task/Instruction: A specific task given to the model.
  - Context: Relevant background information necessary to perform the task.
  - Input data: The input data that is necessary to perform the given task.
  - Output format: The type/format of the task output → LLMs generate text.
- Prompt injection types:
  - **Direct** → Directly insert malicious task/instruction via web/API
  - **Indirect** → Embed malicious prompts in sources analyzed by the LLM e.g., in a webpage or document which is analyzed by the LLM
  - **Stored** → Harmful prompts embedded in an LLM which retains interactions with users and stored until the (conversation) memory is cleared.
  - **Prompt leaking** → Trick the LLM into revealing system prompts or sensitive information from previous prompts → **Discuss:** Is this really diff from above 3?

Prompt injection attacks elaborated 2/2

https://www.invicti.com/blog/web-security/code-injection/

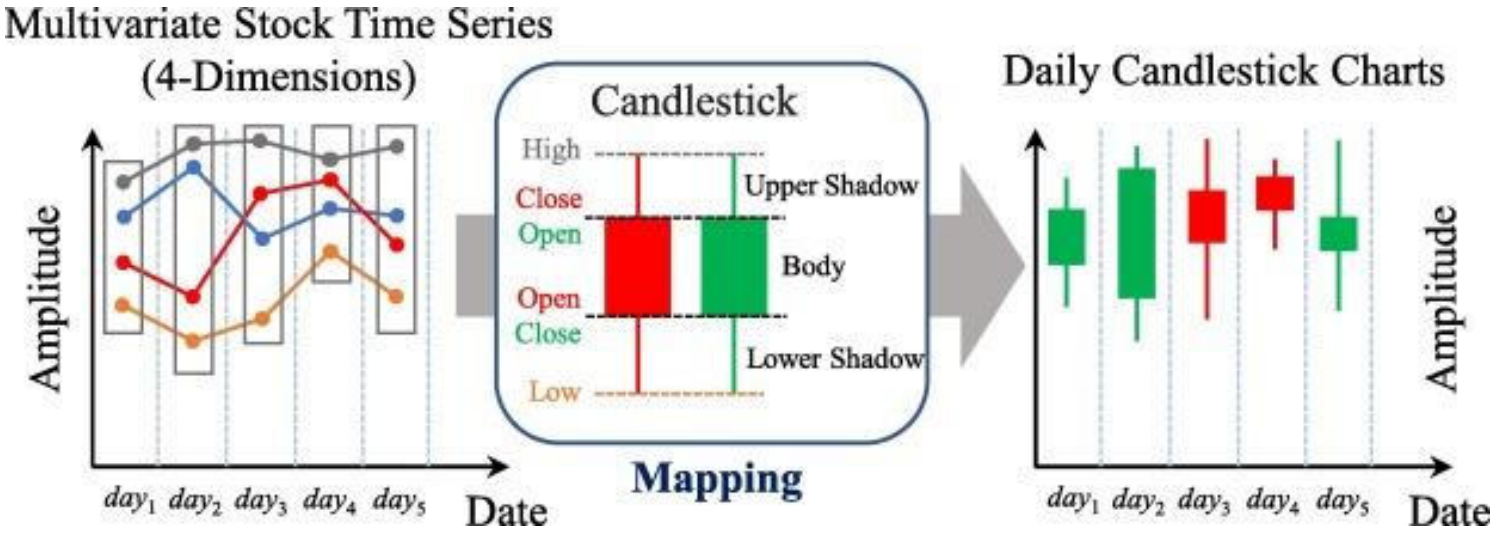
- **Jailbreaking:** Craft prompts which bypass the LLM's safety measures → Output malicious content
- **Data exfiltration:** Craft prompts such that the LLM is convinced into revealing sensitive information
- **Code injection:** Attackers inject source code which is executed by the LLM or its 'connected' systems
- **Persona switching:** Attackers attempt to trick the LLM into changing its assigned persona → Respond to otherwise restricted prompts → Reveal information

ATTACK PATTERNS IN TIME-SERIES DATA



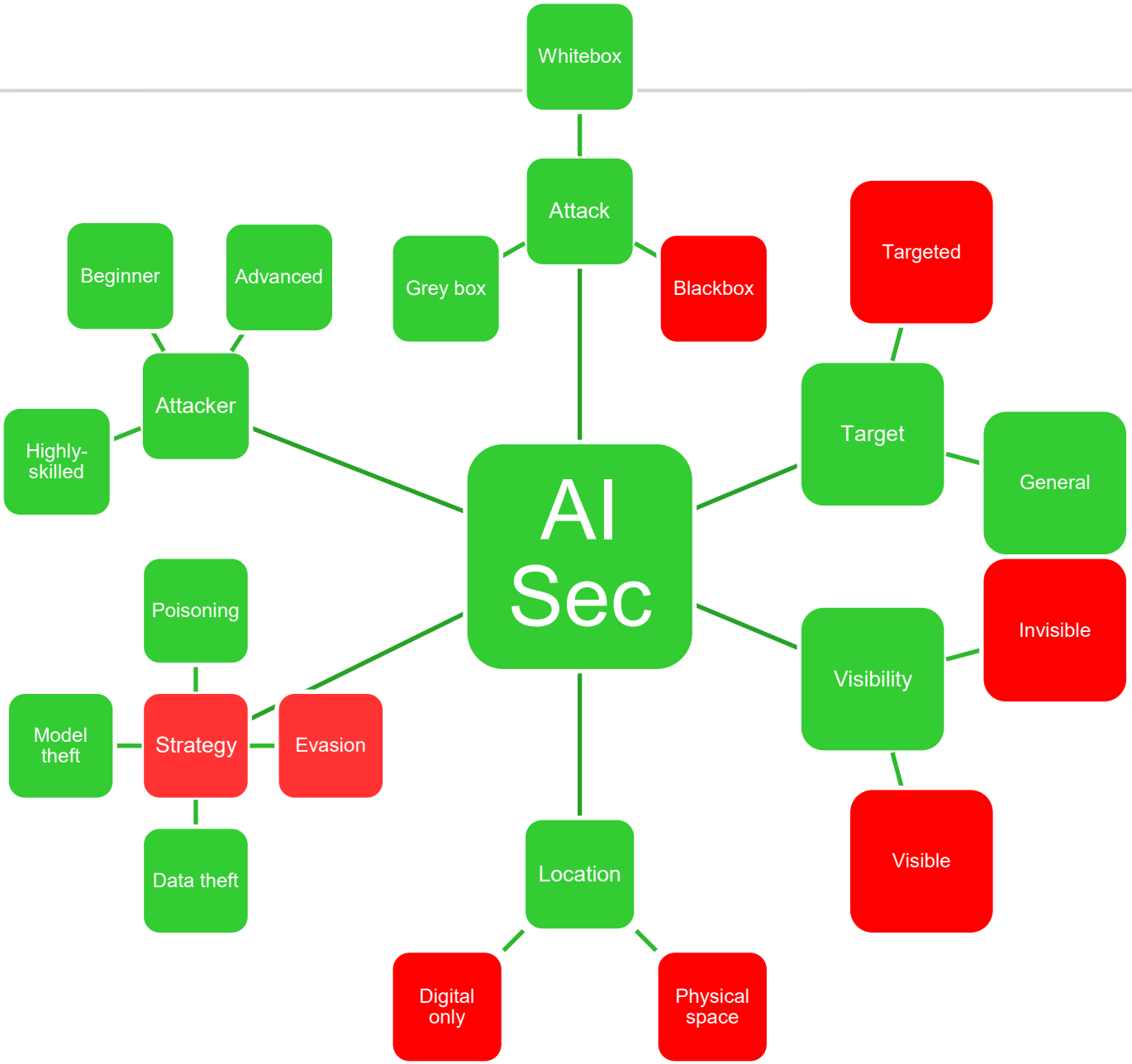
Adversarial samples in time-series data

- Mainly anomaly and change detection algorithms → Not just cybersecurity (!)
- Traditional algorithms: various forms of moving averages (MA); Autoregressive Integrated Moving Average (ARIMA)
  - **Discuss:** How to 'trick' these?
- Latest approaches: Long Short-Term Memory (LSTM) + variants;
  - **Discuss:** How to 'trick' these?



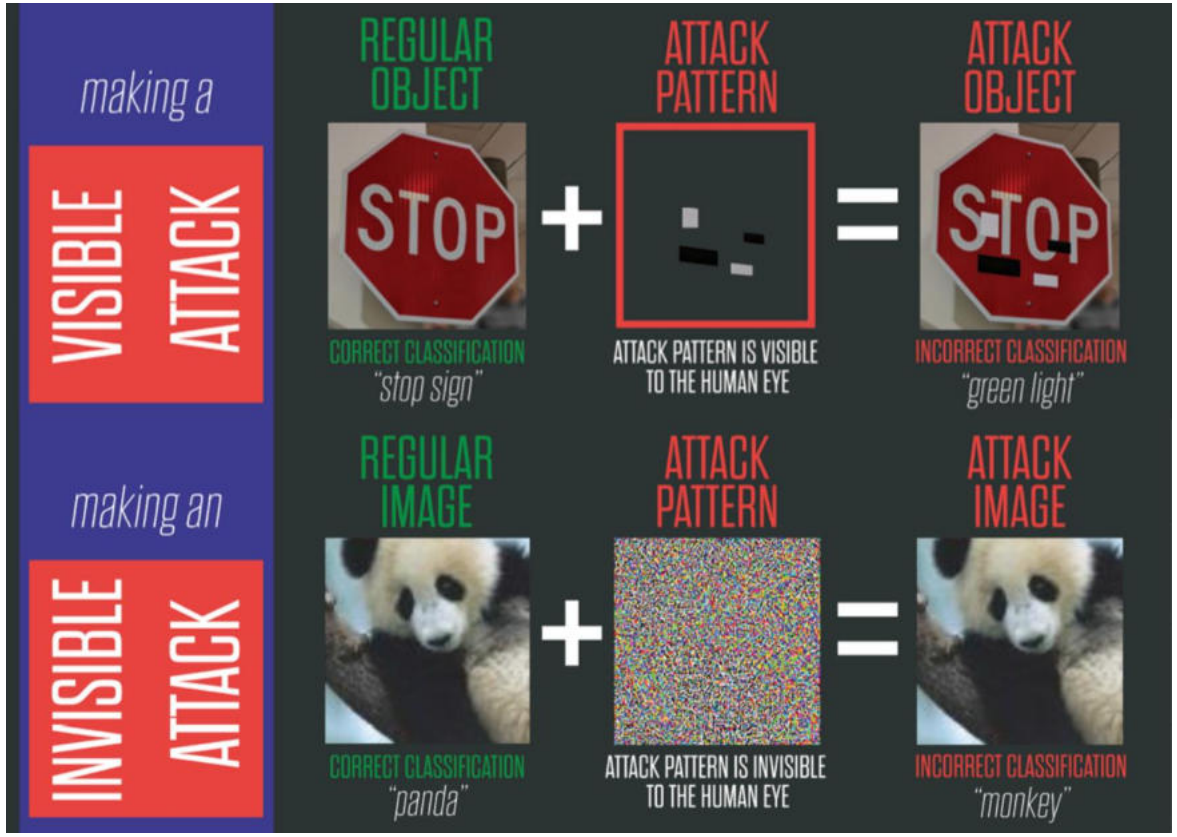
https://www.sciencedirect.com/science/article/pii/S0957417422009071

ATTACK PATTERNS IN MULTIMEDIA



Attack patterns in computer vision

- **ML tasks:** Recognize faces or objects
- **ML model:** Usually deep neural network
- **Known/common attack types:**
  - 1-pixel
  - Visible stickers
  - Targeted noise
  - General noise
- **Visibility:** Visible vs invisible (to the human eye)
- **Location:**
  - Physical space: Stickers on stop signs, fake beard added to face
  - Electronic space: Upload of manipulated images



Copy and move attacks

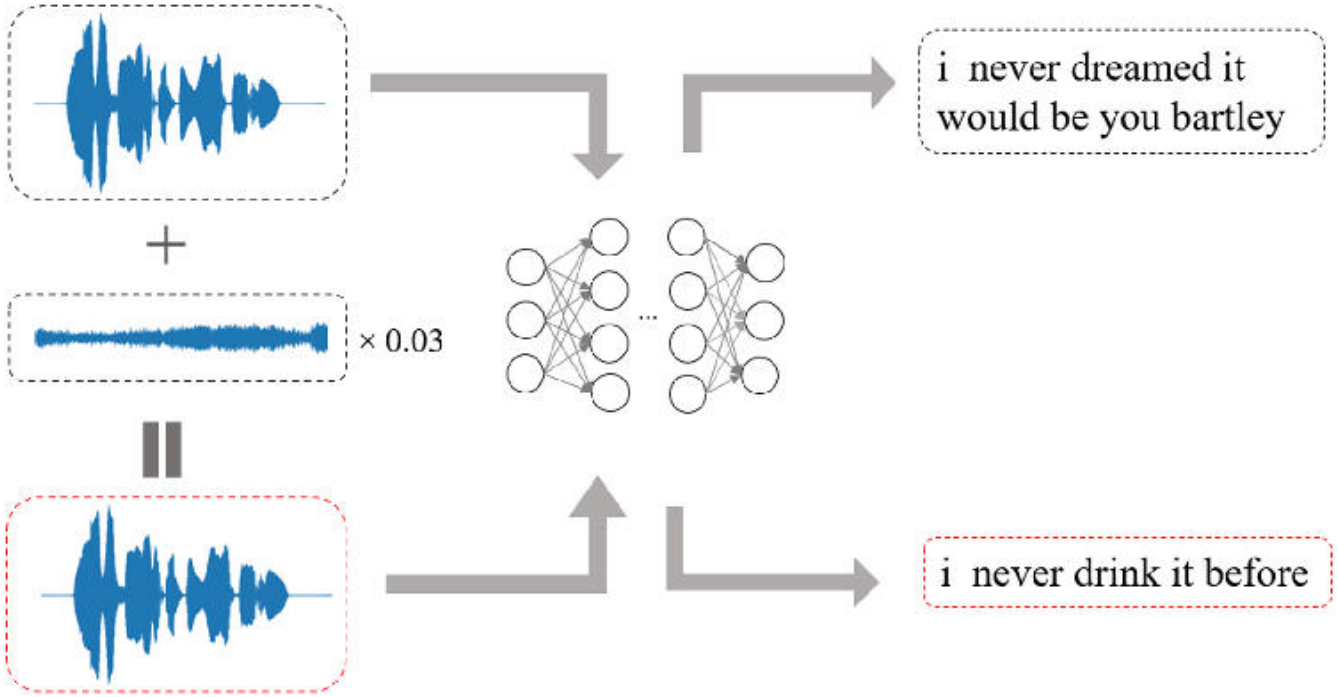
- **ML tasks:** Any image analysis
- **ML model:** Deep neural network
- **Known/common attack types:**
  - Copy one part of image over another area of the same image
  - Copy one part of another image over an area in the analyzed
  - Manipulate the physical space by adding unexpected objects or patterns
- **Visibility:** Usually visible (to the human eye)
- **Location:**
  - Physical space: Manipulated physical space
  - Electronic space: Upload of copy & move manipulated images

(b) Copy-move forgery with geometrical attack (re-scale)

https://www.researchgate.net/publication/327592460_Region_Duplication_Detection_in_Digital_Images_based_on_Centroid_Linkage_Clustering_of_Key-points_and_Graph_Similarity_Matching/figures?lo=1

Attack patterns in audio

- **ML tasks:** Speech to (electronic) text, voice-based authentication
- **ML model:** ???
- **Known/common attack types:**
  - Alter voice to avoid voice-based recognition by police
- **Visibility:** Audible vs inaudible
- **Location:**
  - Physical space: Intentional modification of audio waves in the physical space
  - Electronic space: Upload of manipulated audio

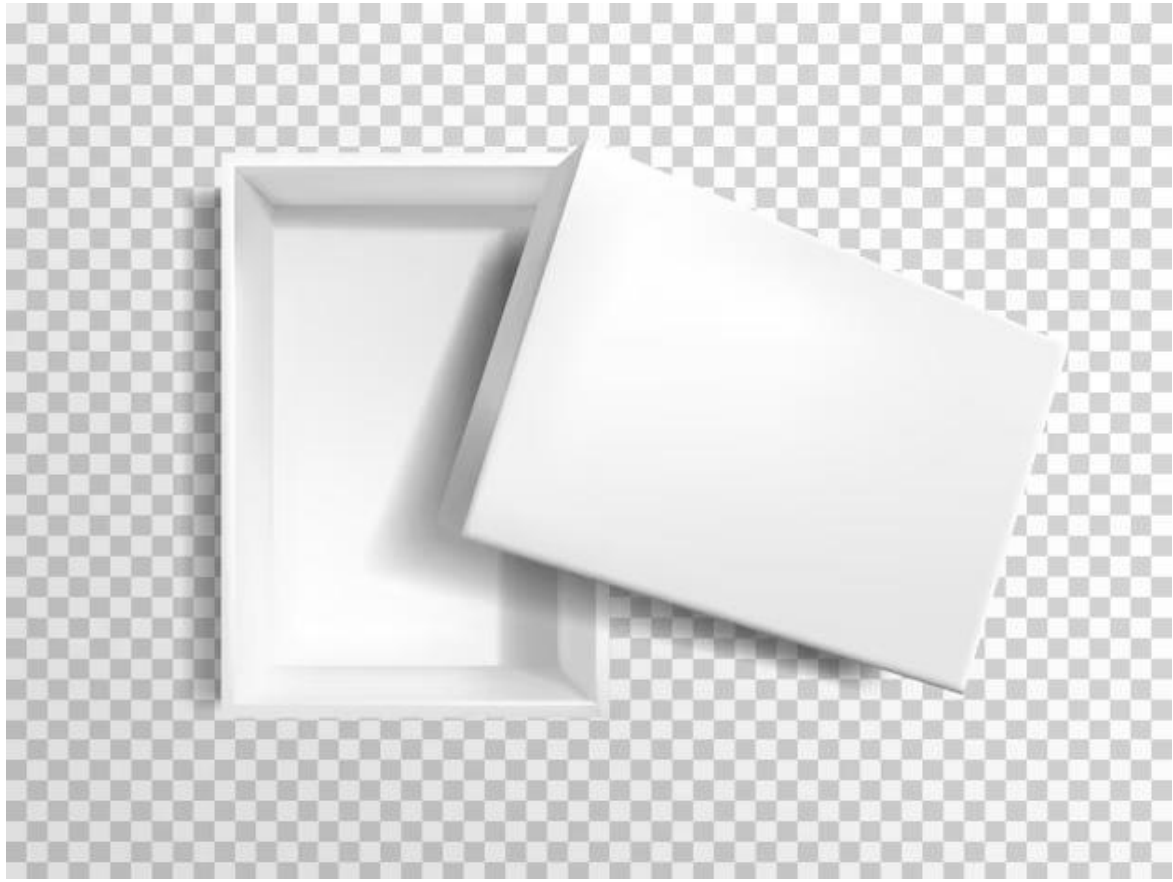


WHITE BOX INPUT DATA MANIPULATION

Intro into white box attacks

- The attacker obtains access to the model
  - Model architecture
  - Model parameters
  - (Optional) Algorithm & hyperparameters
- Training data (usually) not available

- White box attack procedure:
  - Analyze the model
  - (Optional) Design attack → Only if highly skilled attacker
  - Generate adversarial samples
  - Submit inputs to the model
  - Collect and analyze outputs
  - Validate attack results in test
  - Deploy attack against operational AI

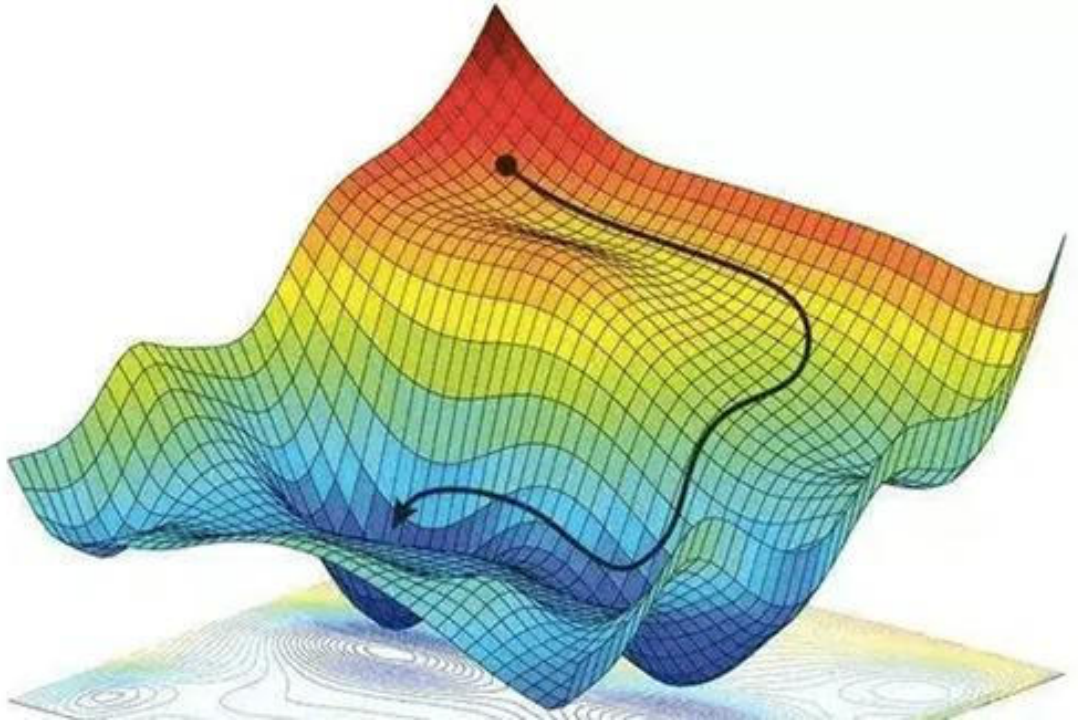


Fast Gradient Sign Method (FGSM)

- **DEF:** The FGSM method generates adversarial input data with the following procedure

$$X^* = X + \epsilon sign\left(\nabla_x J(\theta, X, y)\right)$$

  - X^*: adversarial sample
  - ϵ: perturbation amplitude
  - J: cost function used to train the NN
  - ∇_x: model gradient for sample X and label y
  - θ: model hyperparameters
- The goal of FGSM is to add noise and influence classification decisions → Designed to generate adversarial images
  - The Basic Interaction Method (BIM) and Projected Gradient Descent (PGD) are FGSM variants
- **Discuss:** Are these model architecture specific?



https://easyai.tech/en/ai-definition/gradient-descent/

A. Kurakin, I.J. Goodfellow, S. Bengio, Adversarial examples in the physical world, in: Artificial Intelligence Safety and Security, Chapman and Hall/CRC, 2018, pp. 99–112.
I.J. Goodfellow, J. Shlens, C. Szegedy, Explaining and harnessing adversarial examples, 2014, arXiv preprint arXiv:1412.6572.

Jacobian-based Saliency Map Attack (JSMA)

- **DEF:** The JSMA iterative approach analyzes input feature subset influence on classification decisions
  - Utilizes the model Jacobian to identify gradient

$$J_F = \frac{\delta F(X)}{\delta X}$$

- Note: Allows attackers to analyze the influence of input data features on classification decisions

$$\mathbf{J} = \frac{d\mathbf{f}(\mathbf{x})}{d\mathbf{x}} = \left[\frac{\partial \mathbf{f}(\mathbf{x})}{\partial x_1} \cdots \frac{\partial \mathbf{f}(\mathbf{x})}{\partial x_u}\right] = \begin{bmatrix} \frac{\partial f_1(\mathbf{x})}{\partial x_1} & \cdots & \frac{\partial f_1(\mathbf{x})}{\partial x_u} \\ \vdots & & \vdots \\ \frac{\partial f_v(\mathbf{x})}{\partial x_1} & \cdots & \frac{\partial f_v(\mathbf{x})}{\partial x_u} \end{bmatrix}$$

https://machinelearningmastery.com/a-gentle-introduction-to-the-jacobian/

N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z.B. Celik, A. Swami,The limitations of deep learning in adversarial settings, in: 2016 IEEE European Symposium on Security and Privacy (EuroS&P), IEEE, 2016, pp.372–387.

Carlini & Wagner

- **DEF:** The Carlini & Wagner method generates minimum perturbations which modify classification decisions

$$\min\big(\Delta(x, x+\delta)\big), \text{where}$$
$$C(x+\delta) = t, x+\delta \in [0,1]^n$$
$$x\text{: the sample}$$
$$\delta\text{: the perturbation}$$
$$\Delta\text{: the distance}\{L_0, L_2, L_\infty\}$$

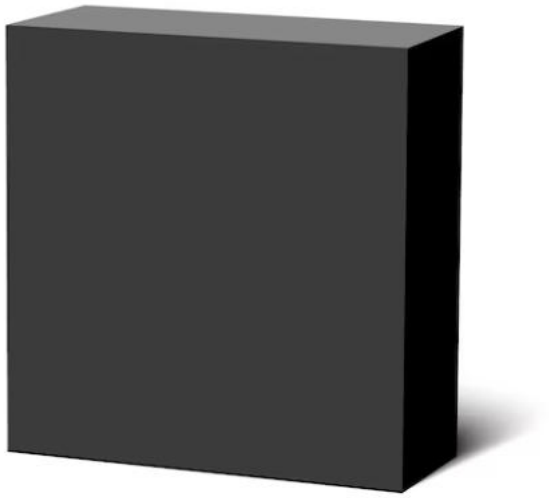
- Usually more efficient compared to FGSM and JSMA due to smaller adversarial sample set

- **Discuss:** Why is the set smaller?

BLACK BOX INPUT DATA MANIPULATION

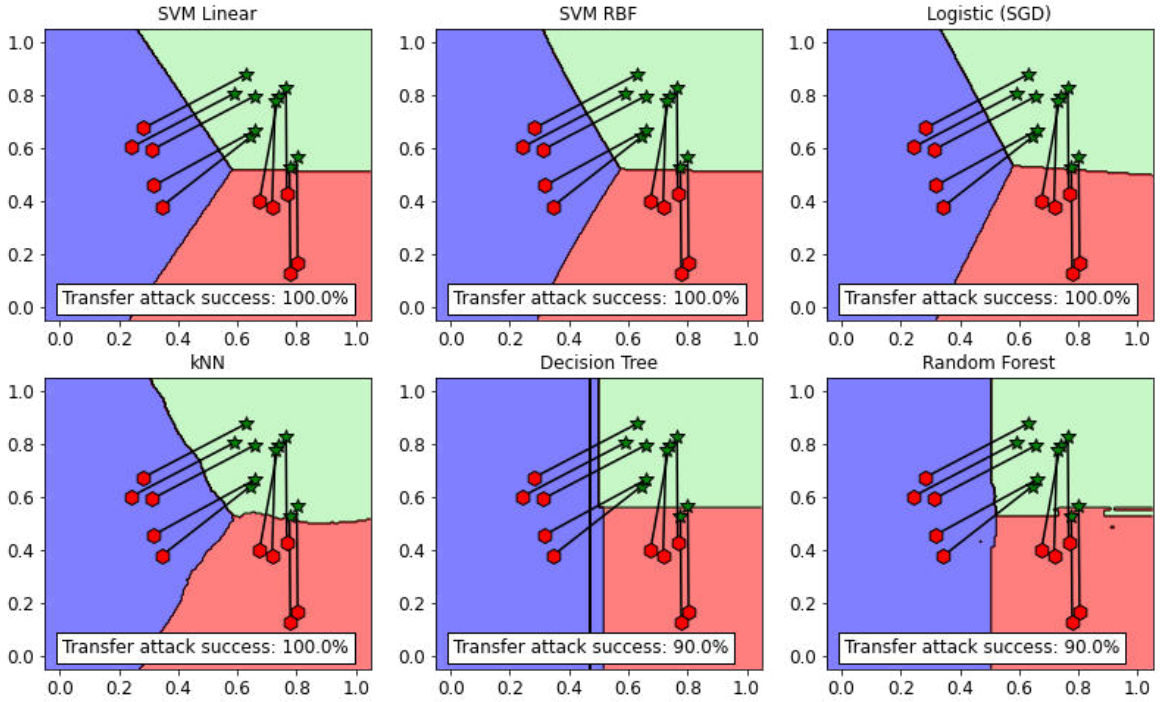
Intro into black box attacks

- The attacker does not have access to the model architecture and parameters, neither the training data
- The attacker obtains black box access to the model
  - Ideally there is no rate limiting on the API or interface used to query the model → **Discuss:** Why?
- Black box attack procedure:
  - (Optional) Design attack → If highly skilled attacker
  - Generate adversarial samples → Generate tailored samples limiting model queries for obtaining required model knowledge
  - Submit adversarial samples to the model
  - Collect and analyze outputs → Observe confidence and other KPIs to further optimize adversarial sample generation
  - Validate attack success
- **Discuss:** What are the attackers' (usual) motives? Note: Refer to the use cases in different sectors discussed!



Transferability of Attacks

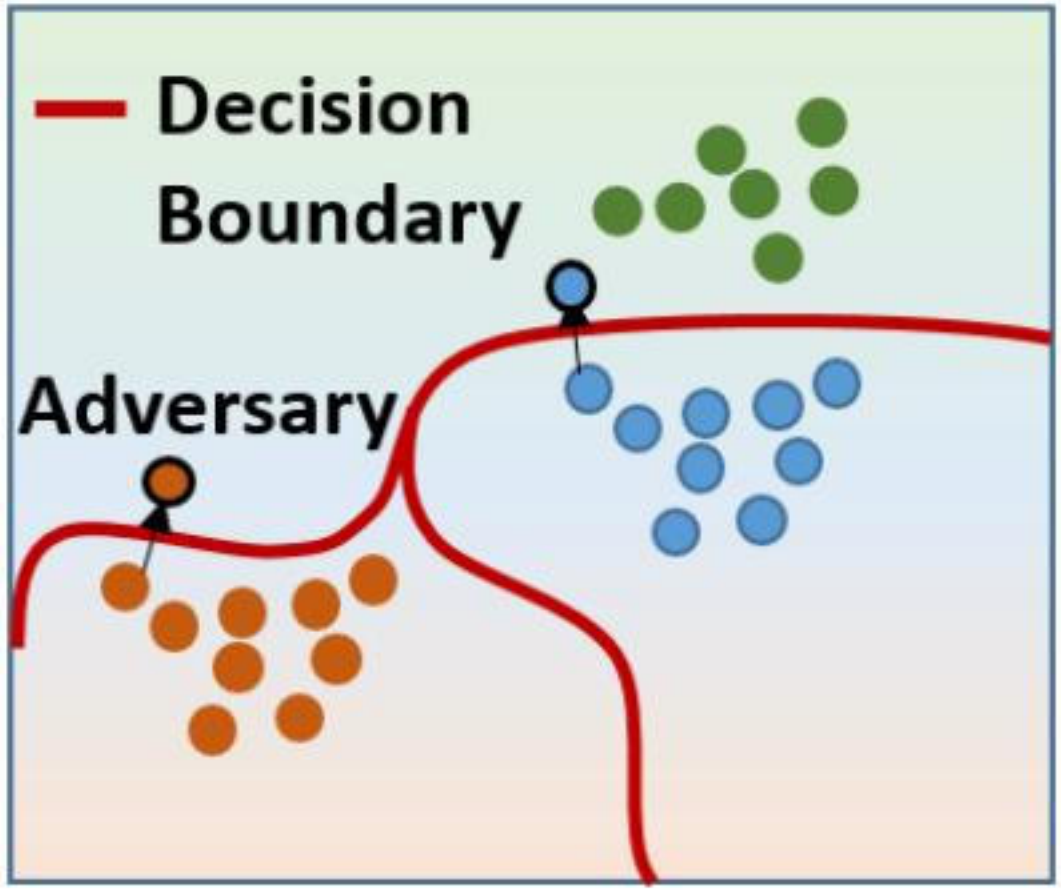
- **Description:** Attacker trains substitute ML model (or ensemble) to test malicious inputs on, then submits inputs to another model
  - Note: The attacker might submit queries to the target model to fine-tune the substitute model training process
  - **Discuss** → Intersecting decision boundaries in benign and adversarial dimensions
- **Stage:** Both, as the attacked AI model is in operation, but the substitute model is trained by the attacker
- **Attack:** Blackbox (in relation to the target AI model)
- **Example:** Scientific references (Papernot and others)
- **AI-specific mitigation:** model ensembles, explainable AI → **Discuss:** Any other?



https://secml.readthedocs.io/en/stable/tutorials/04-Transferability.html

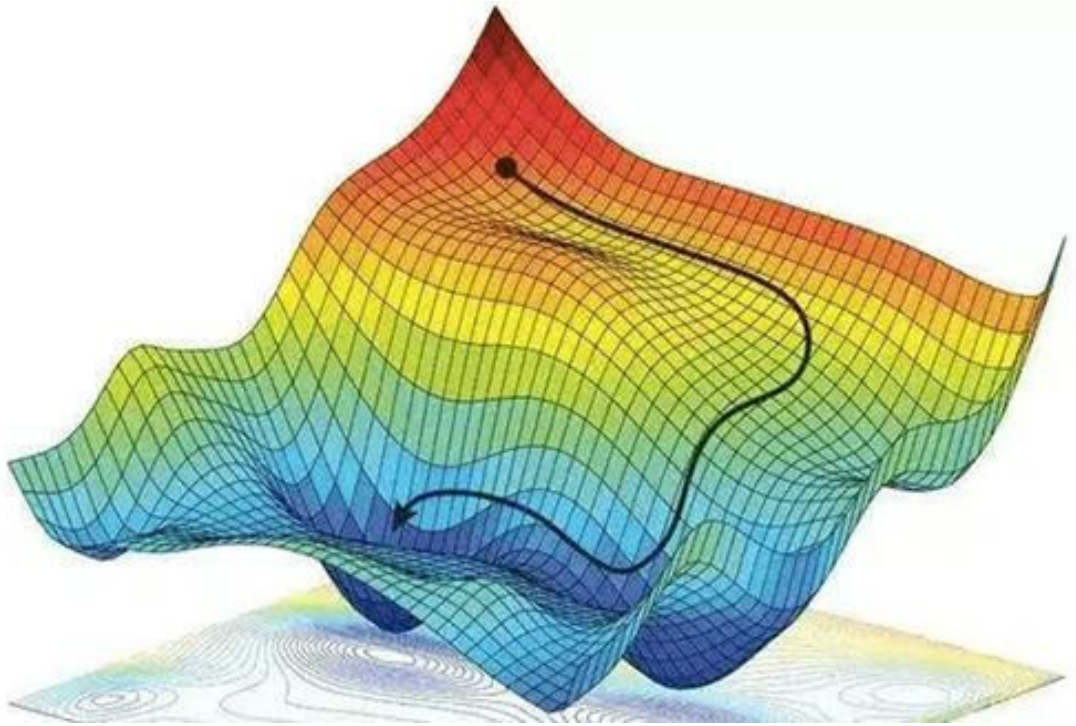
Decision boundary attack

- **DEF:** In a decision boundary attack the attacker maps the model's decision boundaries and crafts malicious input near the boundaries which 'trick' the model into making incorrect decisions
  - Note: Relatively simple to do with tabular data, not so with time-series or images
- The decision boundary is a (hyper)plane or area across which the model's decision changes
  - Classification → Transition between classes
  - Clustering → **Discuss:** Attacker options?
  - **Discuss:** Boundaries in other AI uses?



Gradient estimation

- **DEF:** Gradients are essential in ML training → They indicate how much a model's output changes in response to input changes (gradient descent)
- Gradient estimation process in a black box attack:
  - Send various, cleverly crafted inputs to the AI
  - Observe the impact of inputs on model outputs
  - Approximate model gradients
  - Generate adversarial samples with targeted perturbations based on gradient knowledge
- It is important to minimize the number of necessary queries for gradient estimation
- **Discuss:** Is this really easily doable in real life?



https://easyai.tech/en/ai-definition/gradient-descent/

Exercise: Black & White input manipulation

Task definition & steps

Task definition

- Group A: Prepare a brief report on **white box** input data manipulation
  - Minimum topics: FGSM, PGD, BIM, C&W, JSMA
  - Include one extra topic (own choice!)
- Group B: Prepare a brief report on **black box** input data manipulation
  - Minimum topics: gradient estimation, decision boundary estimation, attack transferability, jailbreaking in LLMs
  - Include one extra topic (own choice!)

Work procedure

- Use an LLM (free tier is sufficient)
- Prompt the LLM to describe the requested 4-6 topics
- Find & include formulae online
- Find/generate additional images
- Support with 10+ scientific references (min 2 for each topic!)
  - Check for hallucinations (!!!)
- Add prompts to report (no page limit)
- Prepare a brief PDF report
- Submit via Moodle
- Get 0-3 extra points

Conclusion

- Input data manipulation in machine learning
  - OWASP/ML01:2023: Input Manipulation Attack
  - Input data manipulation in different input data types
  - Prompt injection in large language models
- Black and white box attacks
- AI-assisted exercise (!)

Additional references (mixed)

- AWS Blog, “Common prompt injection attacks”, https://docs.aws.amazon.com/prescriptive-guidance/latest/llm-prompt-engineering-best-practices/common-attacks.html
- Bell B. et al (2024), “Persistent Classification: Understanding Adversarial Attacks by Studying Decision Boundary Dynamics”, https://arxiv.org/html/2404.08069v1
- Biggio, B., & Roli, F. (2018, October). Wild patterns: Ten years after the rise of adversarial machine learning. In Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (pp. 2154-2156).
- Dalvi N., Domingos P., Mausam, Sanghai S., and Verma D. (2004). Adversarial classification. In Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '04). Association for Computing Machinery, New York, NY, USA, 99–108. https://doi.org/10.1145/1014052.1014066
- Demontis, A., Melis, M., Pintor, M., Jagielski, M., Biggio, B., Oprea, A., ... & Roli, F. (2019). Why do adversarial attacks transfer? explaining transferability of evasion and poisoning attacks. In 28th USENIX security symposium (USENIX security 19) (pp. 321-338).
- Dubrov V. (2023), “Understanding Machine Learning Robustness: Why It Matters and How It Affects Your Models”, https://medium.com/@slavadubrov/understanding-machine-learning-robustness-why-it-matters-and-how-it-affects-your-models-5e2cb5838dab
- Catak F.O. (2020), “Adversarial Machine Learning Mitigation: Adversarial Learning”, https://medium.com/data-science/adversarial-machine-learning-mitigation-adversarial-learning-9ae04133c137
- Guan, Z., Zhang, L., Huang, B. et al. Adaptive hyperparameter optimization for black-box adversarial attack. Int. J. Inf. Secur. 22, 1765–1779 (2023). https://doi.org/10.1007/s10207-023-00716-9
- Haldar, S. (2020), “Gradient-based Adversarial Attacks : An Introduction”, https://medium.com/swlh/gradient-based-adversarial-attacks-an-introduction-526238660dc9
- Investopedia, “Autoregressive Integrated Moving Average (ARIMA) Prediction Model”, https://www.investopedia.com/terms/a/autoregressive-integrated-moving-average-arima.asp
- UC Berkeley Center for Long-Term Cybersecurity, „Adversarial Machine Learning: What? So What? Now What?”, https://www.youtube.com/watch?v=JsklJW01bjc&t=11s
- Wisecube AI, “A Comprehensive Overview of Large Language Models”, https://www.wisecube.ai/blog/a-comprehensive-overview-of-large-language-models/

Additional interesting repositories

- Counterfit, https://github.com/Azure/counterfit/
- SecML: Secure and Explainable Machine Learning in Python, https://github.com/pralab/secml
- SecML Malware, https://github.com/pralab/secml_malware
- Trusted-AI, https://github.com/Trusted-AI
- Zen and the Art of Adversarial Machine Learning, https://github.com/moohax/Talks/

Thank you for your attention!

Eötvös Loránd University (ELTE)
Faculty of Informatics (IK)
Pázmány Péter sétány 1/c
1117 Budapest, Hungary

INTEGRITY ATTACKS

Lecture #6 in Introduction to Data Security

Imre Lendák, PhD, GICSP

2026
Budapest, Hungary

Attack types in the CIA triad…

- ML03:2023 Model Inversion
- ML04:2023 Membership Inference
- ML05:2023 Model Theft

- ML09: Output integrity (?)
- ML10: Model poisoning

- ML02: Data poisoning
- ML06: AI supply chain
- ML07: Transfer learning
- ML08: Model skewing
- ML09: Output integrity
- ML10: Model poisoning

- **Discuss:** What about **ML01**??? Is it really an integrity attack?

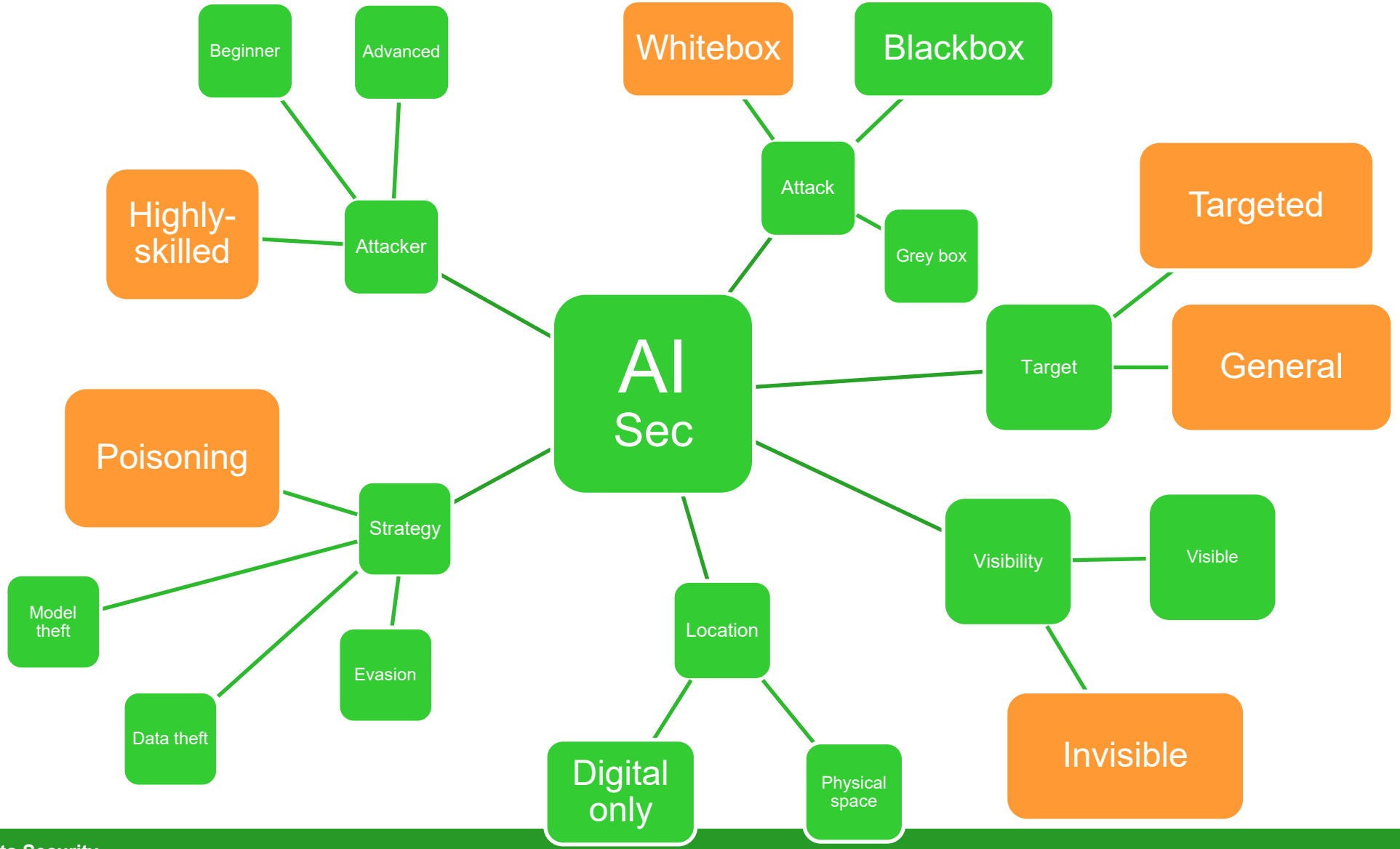
Integrity attacks from different angles

- **DEF:** Integrity attacks are unauthorized modifications of any aspect of the AI operations pipeline (data, algorithm, hyperparameters, model, libraries).
- **Attacker target:** Model vs (training) data → Our main method of dividing this material
- **Application domain:** Computer vision, cybersecurity (spam, malware, intrusion detection), healthcare, finances, natural language processing, LLMs & agentic AI
- **Attacker goal:** Availability or integrity violation (altered model) → Essentially all poisoning attacks violate integrity → **Discuss!**
- **Targeting:** Targeted vs untargeted. Backdoor attacks are targeted & involve a smaller number of samples, while availability poisoning degrades AI performance overall → it is untargeted!

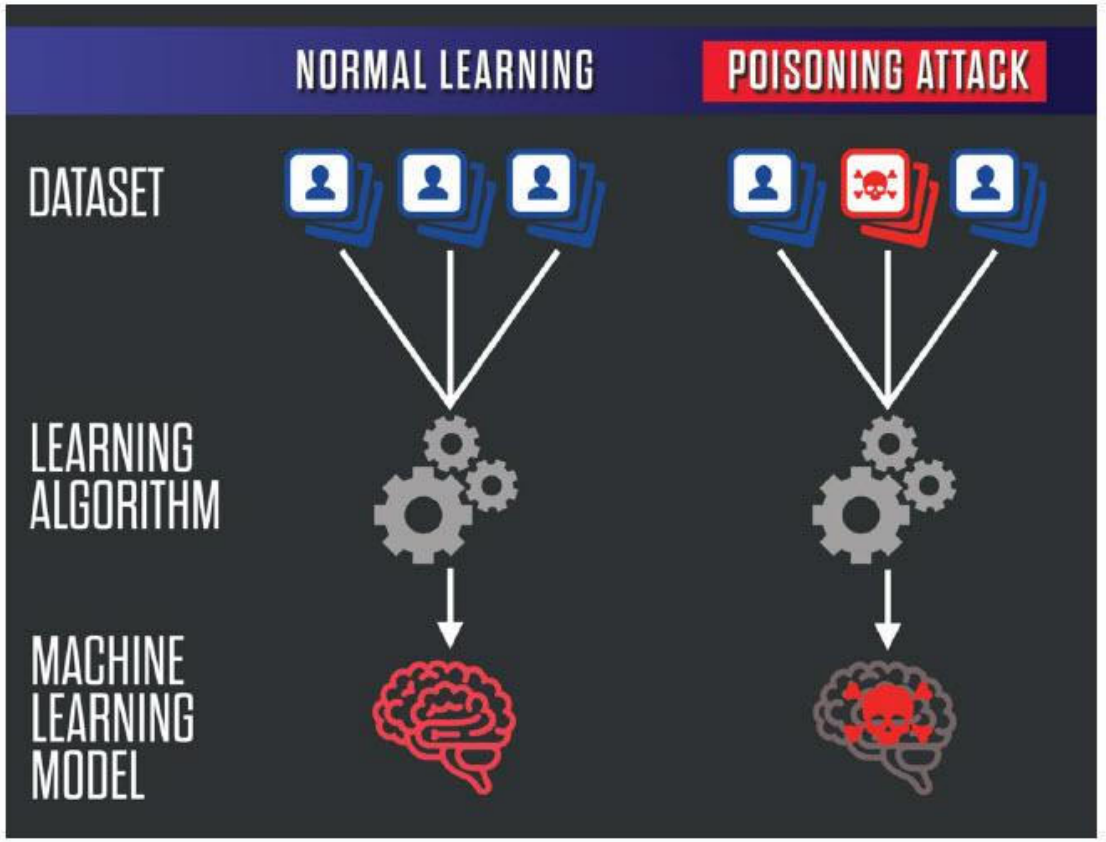
Presentation outline

- Training data poisoning
  - ML02:2023 Data Poisoning Attack
  - ML08:2023 Model Skewing
- Model poisoning
  - ML10:2023 Model Poisoning
  - ML07:2023: Transfer Learning Attack
- Supply chain attacks
  - ML06:2023: AI Supply Chain Attacks
- Other integrity attacks
  - ML09:2023: Output Integrity Attack

Position poisoning attacks in the ontology



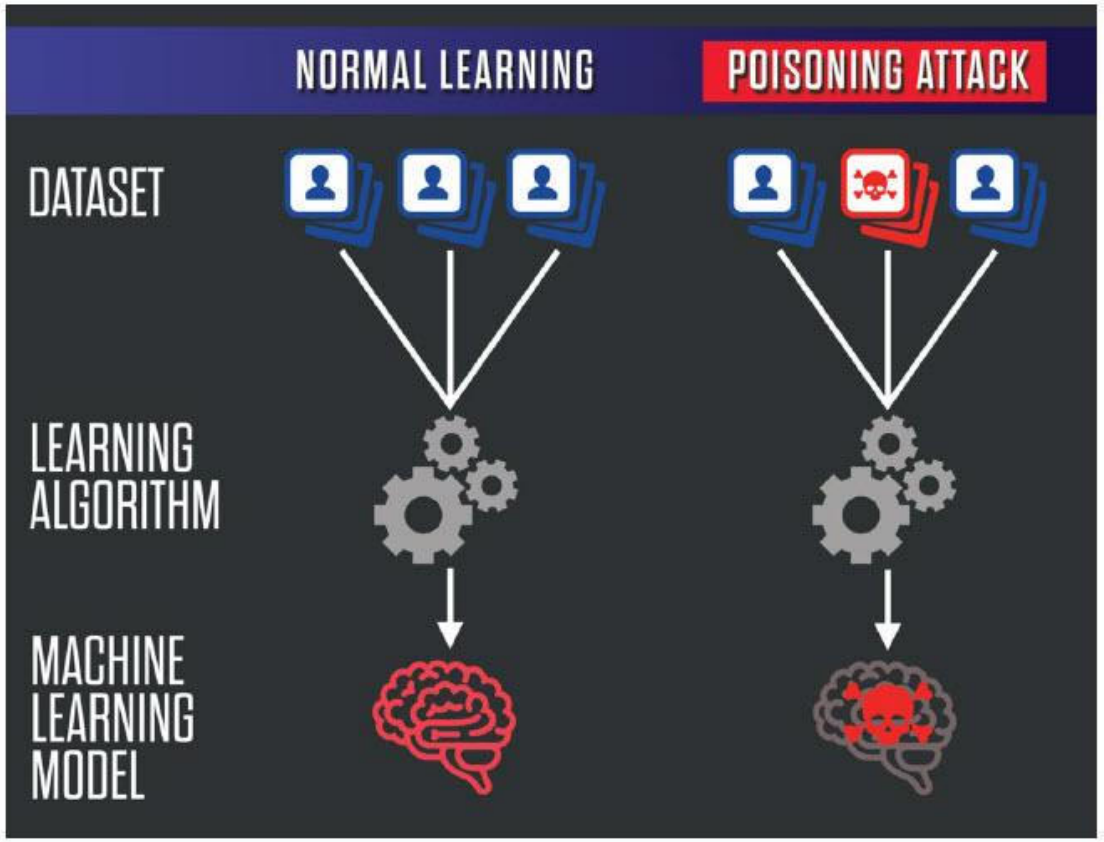
Training Data poisoning



https://informationmatters.net/data-poisoning-ai/

ML02:2023: Data Poisoning Attack

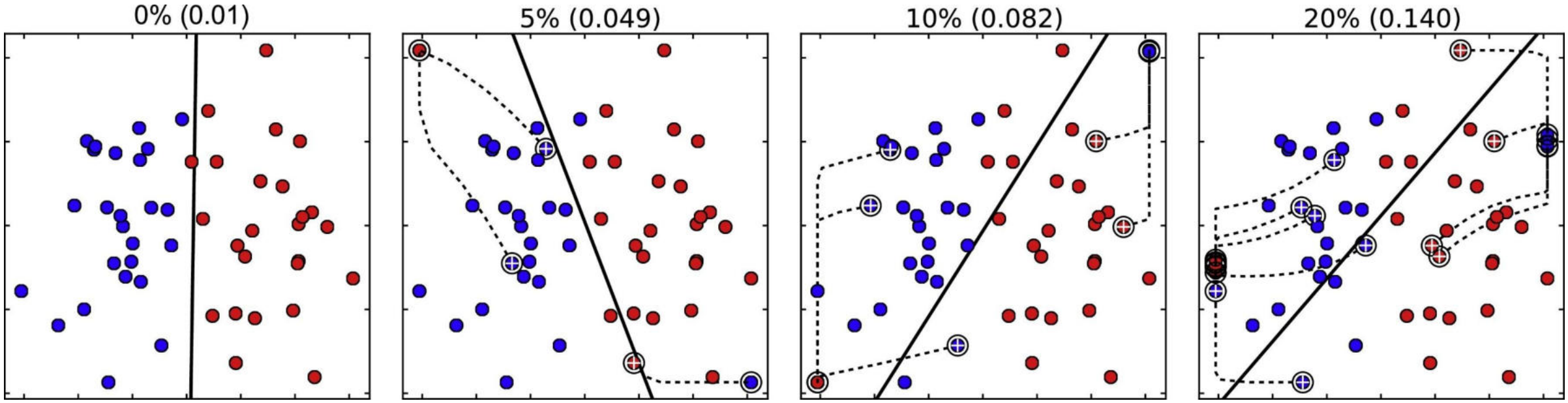
- **Description:** The attacker manipulates the training data to achieve their goals → might be a preparation of an evasion attack (ML01!)
- **Attack:** (usually) whitebox or grey box → **Discuss:** Can it be black box?
- **Stage:** Training phase
- **Source:** Compromise training data storage || malicious insider
- **Example(s):**
  - Insert mis-labeled (email) spam data → Avoid spam detection (i.e., evasion)
  - Nightshade & similar → Intentional poisoning of digital art (!)
- **AI-specific mitigation:**
  - Training data validation
  - Model validation & ensembles
  - Anomaly detection



https://informationmatters.net/data-poisoning-ai/

Poisoning primer on classification

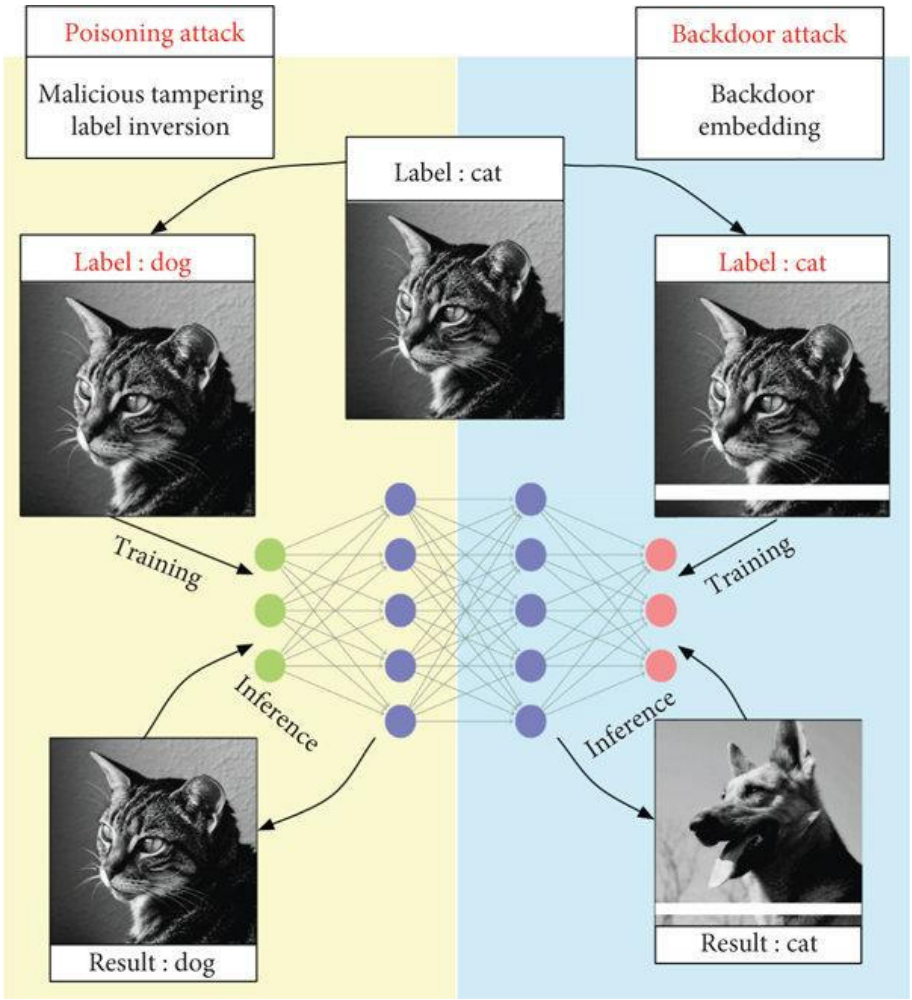
- Label flipping → Alter the label of a subset of data points (observations)
- Clean label attack → Discuss! → Label not modified, only observations (!)
- The attacker can also generate additional data points (!)



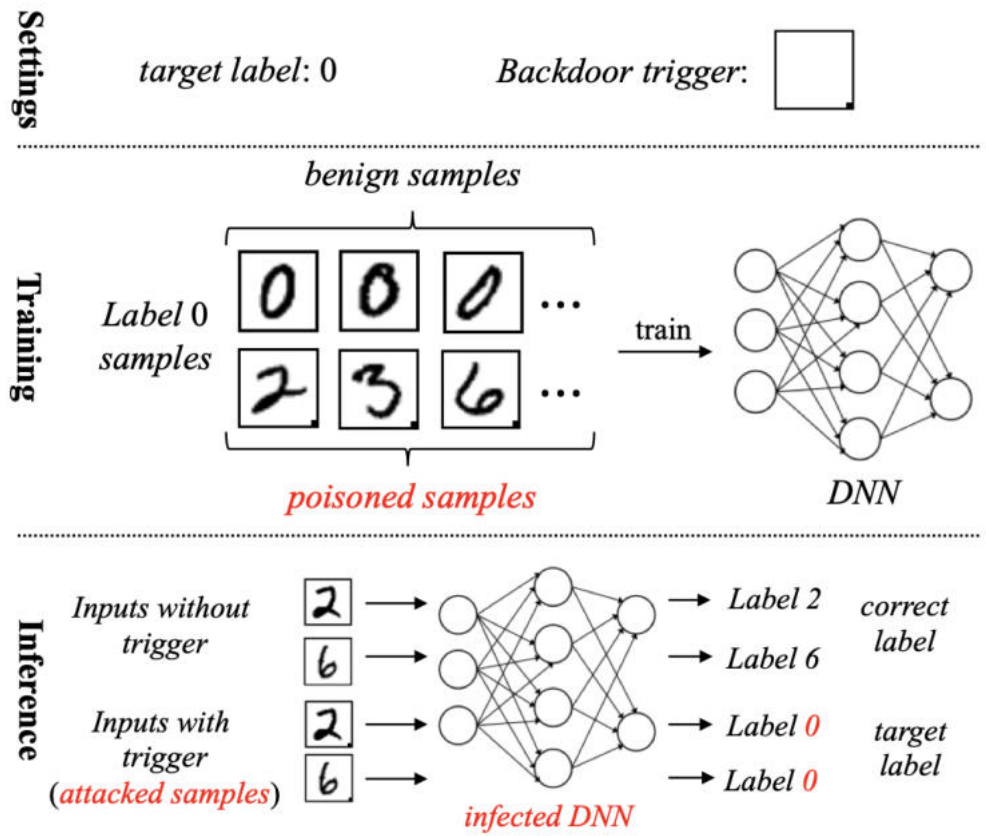
B. Biggio and F. Roli, "Wild patterns: Ten years after the rise of adversarial machine learning," Pattern Recognition, vol. 84, pp. 317–331, 2018.

Poisoning in image analysis – Label flipping

- **Description:** The attacker modifies the labels in the training data to achieve their goals
- **Goal:** Evasion, decreased model performance
- **Attacker:** Sophisticated
  - NOTE: Simple attack, but not easy to obtain necessary access to the training environment
- **Type:** White or grey box
- **Examples:** Evasion attacks (?), degraded model performance



Data backdoor attacks



https://velog.io/@tjdcjffff/Backdoor-Attack

- **Description:** The attacker modifies the training data to achieve their goals
- **Goal:** Evasion, mis-classification → Decreased model performance
- **Attacker:** Sophisticated
- **Attack:** White or grey box
- **Examples:** (???)

Poisoning attacks against LLMs

- **DEF:** In general, prompts consist of the following elements:
  - Task/Instruction: A specific task given to the model.
  - Context: Relevant background information necessary to perform the task.
  - Input data: The input data that is necessary to perform the given task.
  - Output format: The type/format of the task output → LLMs generate text.
- Prompt injection types:
  - **Direct** → Directly insert malicious task/instruction via web/API
  - **Indirect** → Embed malicious prompts in sources analyzed by the LLM e.g., in a webpage or document which is analyzed by the LLM
  - **Stored** → Harmful prompts embedded in an LLM which retains interactions with users and stored until the (conversation) memory is cleared.
  - **Prompt leaking** → Trick the LLM into revealing system prompts or sensitive information from previous prompts → **Discuss:** Is this really diff from above 3?

ML08:2023: Model Skewing

- **Description:** The attacker modifies training data distribution → The model behaves not as planned.
  - **Discuss:** Relation to data poisoning?
- **Stage:** Training
- **Example** (nothing proven in the wild):
  - **Discuss:** Credit rating, criminal rating (?)
- **AI-specific mitigation:**
  - Model IO and performance monitoring, as well as anomaly detection
  - Model re-training
  - Model robustness testing → Testing model sensitivity to specific training data changes

https://www.freepik.com/free-vector/file-transfer-concept-illustration_5843118.htm

MODEL POISONING

https://www.freepik.com/free-vector/file-transfer-concept-illustration_5843118.htm

Model poisoning in (near) real time…

- Tay was Microsoft's Twitter chatbot released on March 23, 2016.
  - Tay = Thinking about you
  - Tay was an online person similar to a 19-year woman from the USA
- Tay was switched off within 16 hours → Racist and aggressive answers caused by trolls/bots interacting with it
- Microsoft official explanation stated that Tay was learning from its interactions → **Discuss:** Was this a sponge training attack?
  - Trolls fed large volumes of malicious prompts into Tay
- **Trivia:** Zo was Tay's successor and it was online between Dec 2016 and May 2019 on multiple platforms (Twitter and others)

https://en.wikipedia.org/wiki/Tay_(chatbot)

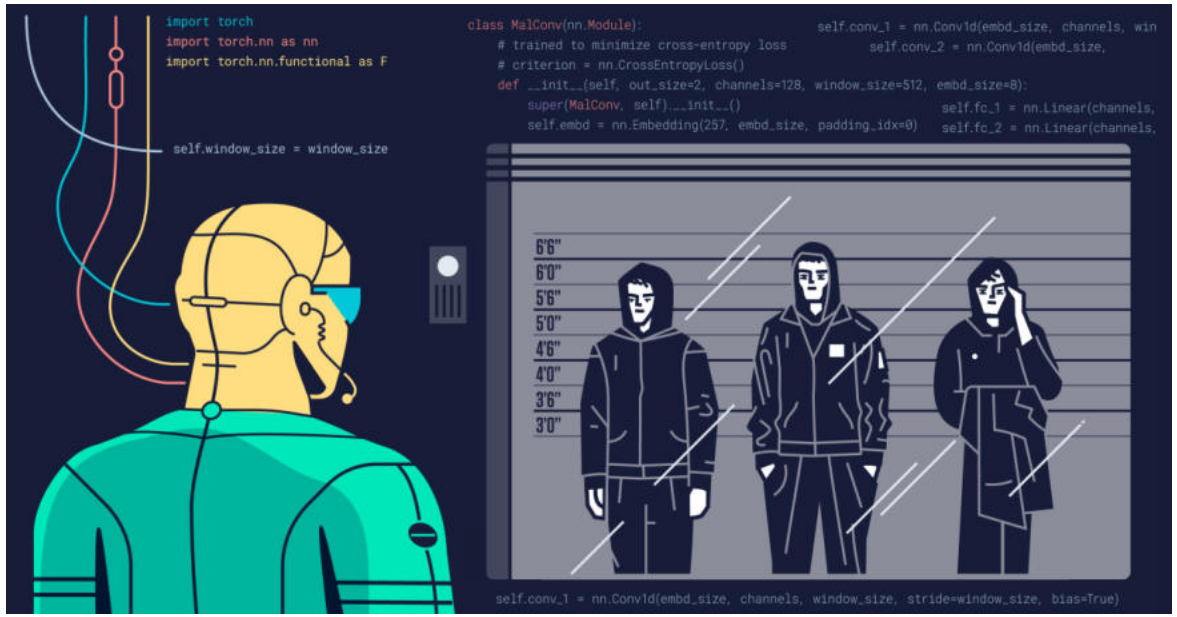
ML10:2023: Model Poisoning

- **Description:** The attacker obtains access to and modifies the AI model parameters.
- **Stage:** AI model use/exploitation
- **Example** (nothing proven in the wild):
  - Credit risk, disease diagnosis, anomaly detection (as in spam and IDS)
- **AI-specific mitigation:**
  - Model regularization
  - Use of robust models
  - Use of model ensembles → **Discuss:** Which are the edge cases in which this helps?

https://www.freepik.com/free-vector/file-transfer-concept-illustration_5843118.htm

Algorithm attack

- **Description:** The attacker modifies the algorithm used in the model training phase
  - Directly tampers with the algorithm code or modifies a linked library
- **Goal:** Evasion, decreased model performance, energy consumption
  - **Discuss:** Other attacker motivation(s)?
- **Attacker:** Sophisticated
  - Can be industrial competitor
- **Attack:** White box
- **Examples:** ???

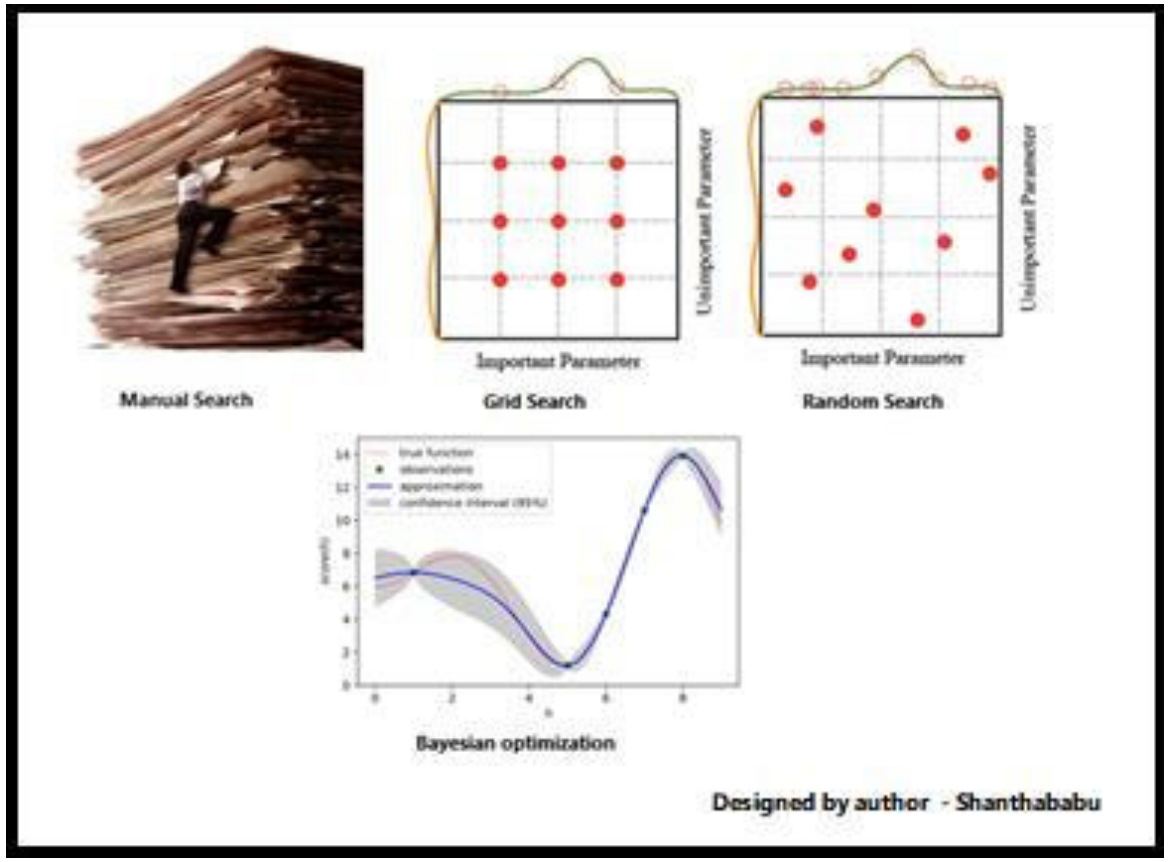


https://www.reversinglabs.com/blog/how-to-harden-ml-models-against-adversarial-attacks

Hyperparameter attack

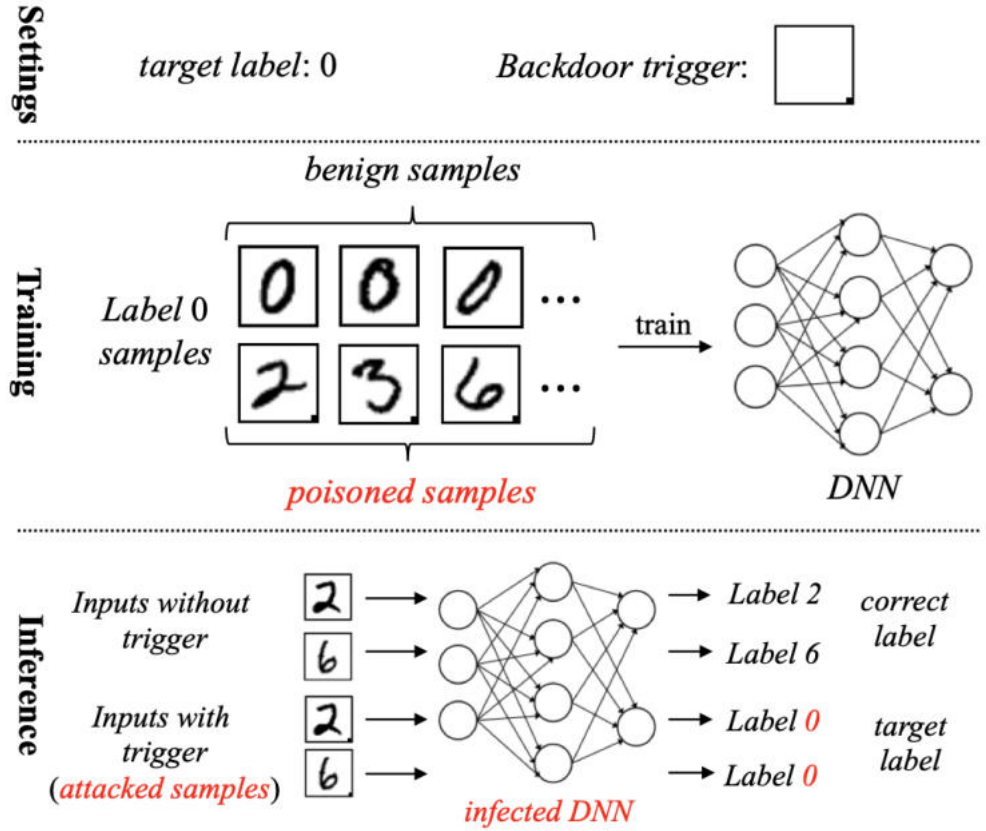
- **Description:** The attacker modifies the hyperparameters used in the training stage
- **Goal:** Evasion, decreased model performance, energy consumption
  - **Discuss:** Other attacker motivation(s)?
- **Attacker:** Sophisticated
- **Attack:** White box
- **Examples:** ???

- **Discuss:** Can hyper parameter search be targeted as well? How?



https://www.datasciencecentral.com/hyperparameter-tuning-techniques-in-machine-learning-engineering/

Model backdoor attacks

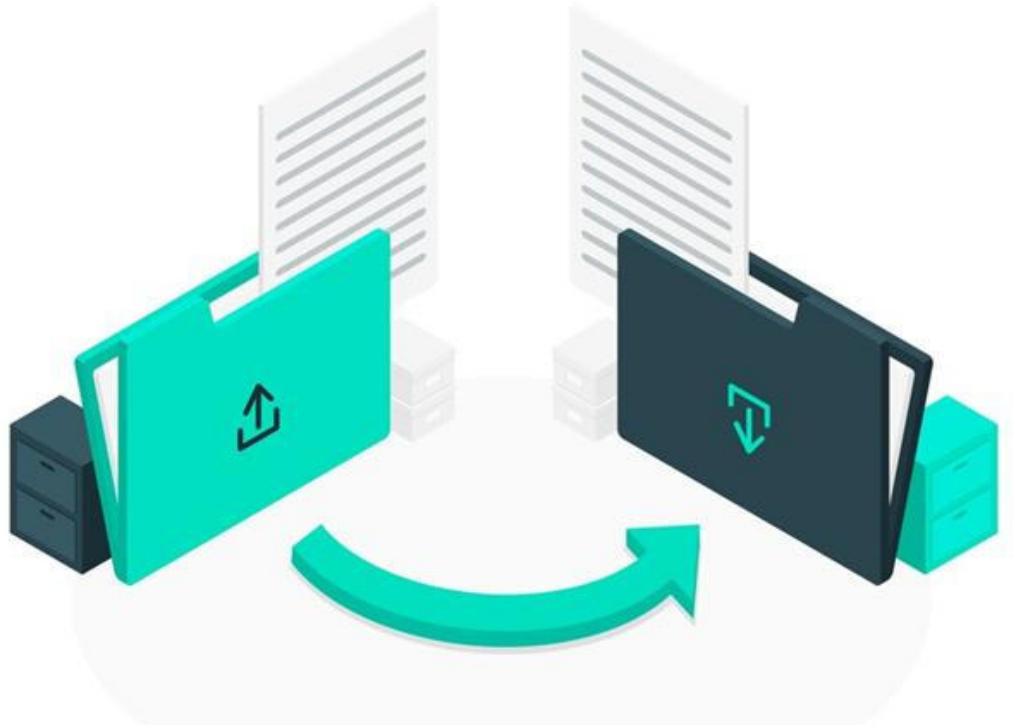


https://velog.io/@tjdcjffff/Backdoor-Attack

- **Description:** The attacker modifies the trained model to achieve their goals
- **Goal:** Evasion, decreased model performance, energy consumption, loss of availability
- **Attacker:** Sophisticated
- **Attack:** White box
- **Examples:** ???

ML07:2023: Transfer Learning Attack

- **Definition:** Transfer learning = A model trained on one task is reused as a starting point for a model on a second task
- **Risks:** Model adoption can introduce risks/malicious actors might modify starting (foundation) model.
- **Stage:** Both stages, but the desired impact is usually in exploitation
- **Example** (nothing proven in the wild):
  - The attackers alters frequently used ‘base’ model → **Discuss:** Relation to model poisoning under ML10?
- **AI-specific mitigation:**
  - Model IO monitoring and anomaly detection → **Discuss:** How and what to measure?

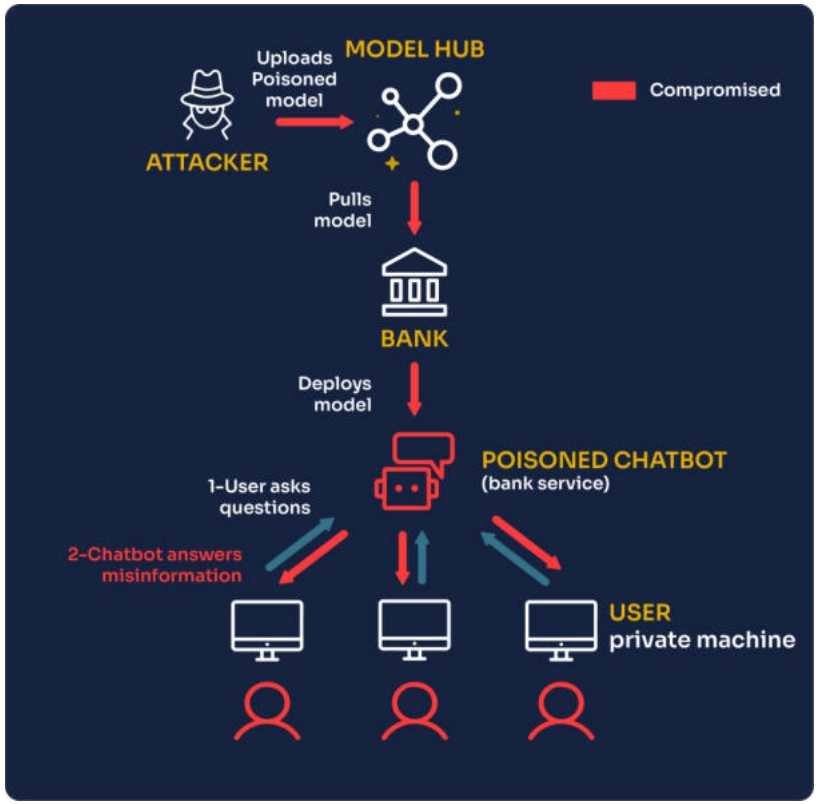


https://www.freepik.com/free-vector/file-transfer-concept-illustration_5843118.htm

SUPPLY CHAIN RISKS

ML06:2023: AI Supply Chain Attacks

- **Description:** The attacker targets the supply chain of AI operators. Supply chain elements: data/model hubs, model mgmt., MLOps libs & tools.
  - **Discuss** → How to separate from data and model poisoninig?
- **Stage:** Both training and exploitation
- **Example** (nothing proven in the wild):
  - The attackers alters upstream data, models or MLOps platforms
- **AI-specific mitigation:**
  - Model IO monitoring and anomaly detection → **Discuss:** How and what to measure?



https://blog.mithrilsecurity.io/attacks-on-ai-models-prompt-injection-vs-supply-chain-poisoning/

What constitutes the AI supply chain?

- Data:
  - Data on the web: text, images
  - Datasets hubs (Kaggle?)
- Model:
  - Foundation models
  - Model brokers
  - AI superscalers → **Discuss:** Which models are available for download?
- Training:
  - 'Model-building' libraries used
  - Algorithms
  - Blogs about model tuning
- Operations:
  - AIops libraries used
- **Discuss** → Which are the specific risks related to each of these categories?
- **Discuss** → Anything else?

OTHER INTEGRITY ATTACKS

ML09:2023: Output Integrity Attack

- **Description:** The attacker manipulates the output of an AI system to reach their goals.
- **Stage:** AI model use/exploitation
- **Example** (nothing proven in the wild):
  - Idea: Modify output of disease diagnosis AI → Wrong prescription with dire consequences
- **AI-specific mitigation:**
  - Model IO and performance monitoring, as well as anomaly detection

https://www.freepik.com/free-vector/file-transfer-concept-illustration_5843118.htm

SUMMARY AND REFERENCES

Summary and next steps

- Training data poisoning
  - ML02:2023 Data Poisoning Attack
  - ML08:2023 Model Skewing
- Model poisoning
  - ML10:2023 Model Poisoning
  - ML07:2023: Transfer Learning Attack
- Supply chain attacks
  - ML06:2023: AI Supply Chain Attacks
- Other integrity attacks
  - ML09:2023: Output Integrity Attack

Additional references – 1

- Adam Lundquist, “Backdoor Attacks on AI Models”, https://www.cobalt.io/blog/backdoor-attacks-on-ai-models
- Data Science Central, “Hyperparameter Tuning Techniques in Machine Learning Engineering”, https://www.datasciencecentral.com/hyperparameter-tuning-techniques-in-machine-learning-engineering/
- Information Matters, “Data Poisoning: a Ticking Time Bomb”, https://informationmatters.net/data-poisoning-ai/
- Kumar Danush, “LLM poisoning”, https://medium.com/@danushidk507/llm-poisoning-44ddec486010
- LeewayHertz, „Prioritizing security in AI development: Training, building, and deploying models in a secure environment”, https://www.leewayhertz.com/security-in-ai-development/
- Rod Trent (2023), “Must Learn AI Security Part 10: Backdoor Attacks Against AI”, https://rodtrent.substack.com/p/must-learn-ai-security-part-10-backdoor
- Rod Trent (2023), “Must Learn AI Security Part 2: Data Poisoning Attacks Against AI”, https://rodtrent.substack.com/p/must-learn-ai-security-part-2-data

Thank you for your attention!

Eötvös Loránd University (ELTE)
Faculty of Informatics (IK)
Pázmány Péter sétány 1/c
1117 Budapest, Hungary

CONFIDENTIALITY ATTACKS AGAINST AI

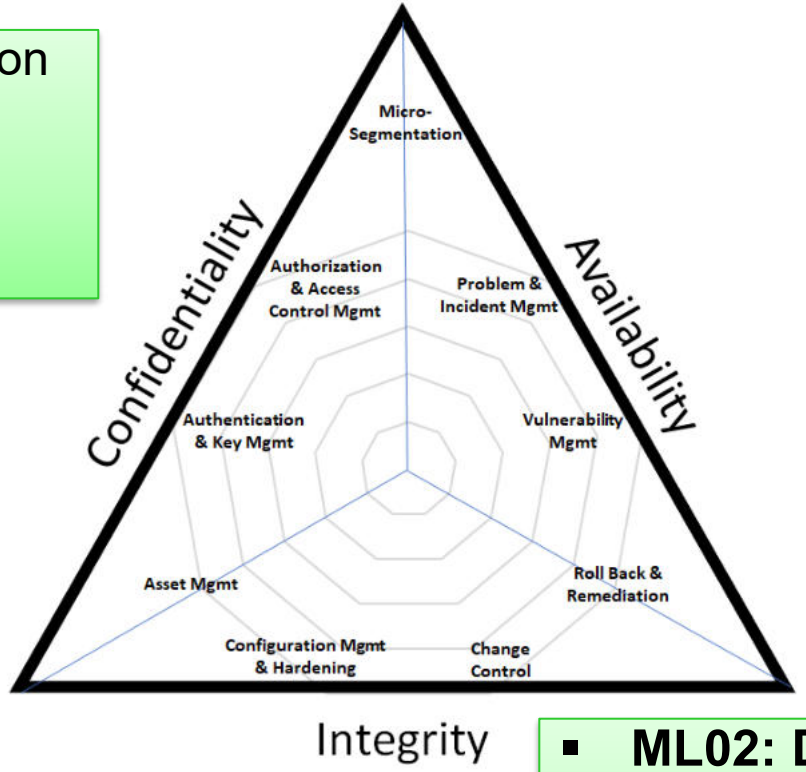
Course: Introduction to Data Security

Imre Lendák, PhD, GICSP

2026
Budapest, Hungary

Attack types in the CIA triad…

- ML03:2023 Model Inversion
- ML04:2023 Membership Inference
- ML05:2023 Model Theft



- ML09: Output integrity (?)
- **ML10: Model poisoning**

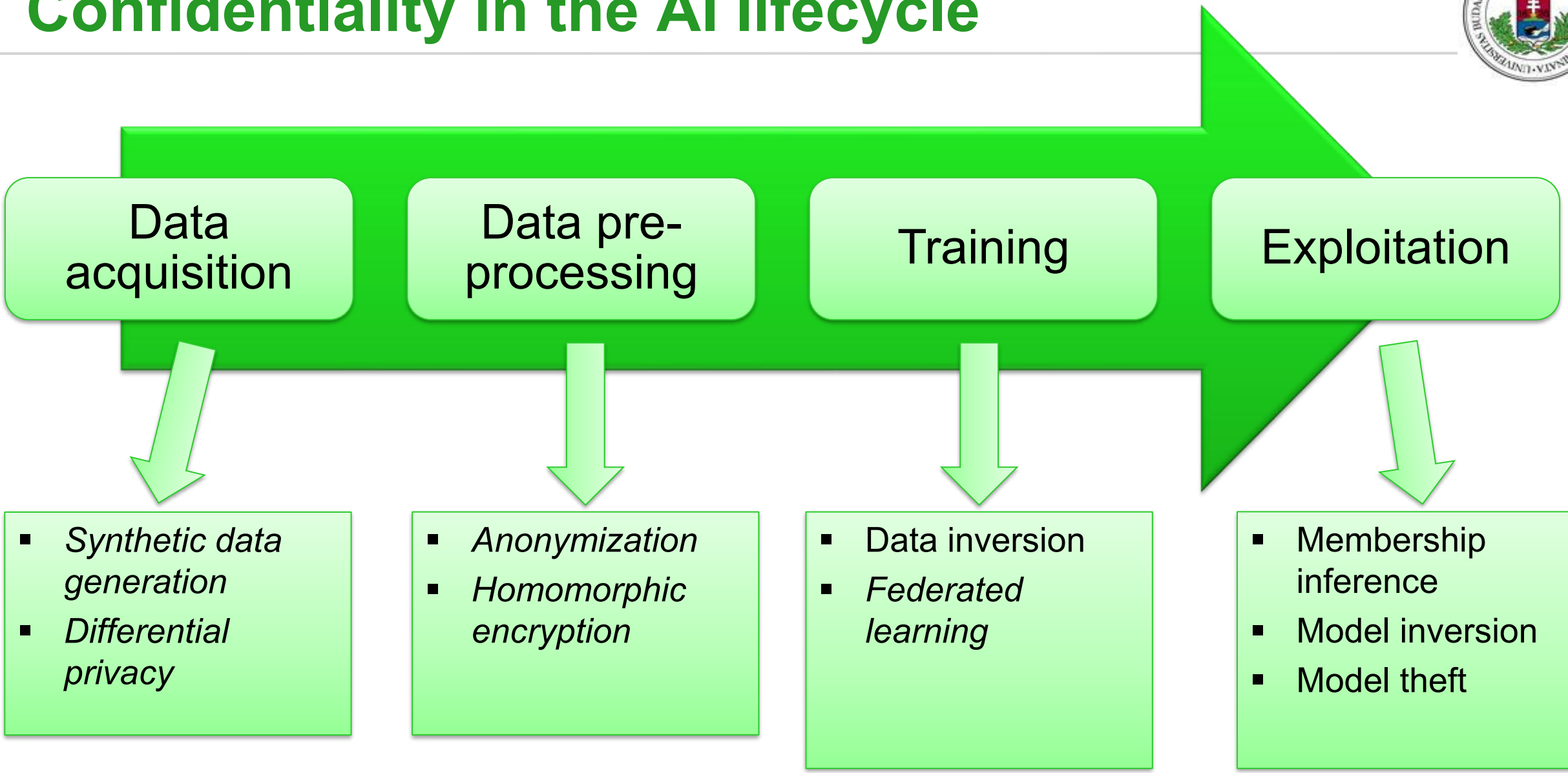
- **Discuss:** What about **ML01**??? Is it really an integrity attack?

- **ML02: Data poisoning**
- **ML06: AI supply chain**
- ML07: Transfer learning
- ML08: Model skewing
- ML09: Output integrity (?)
- **ML10: Model poisoning**

Presentation outline

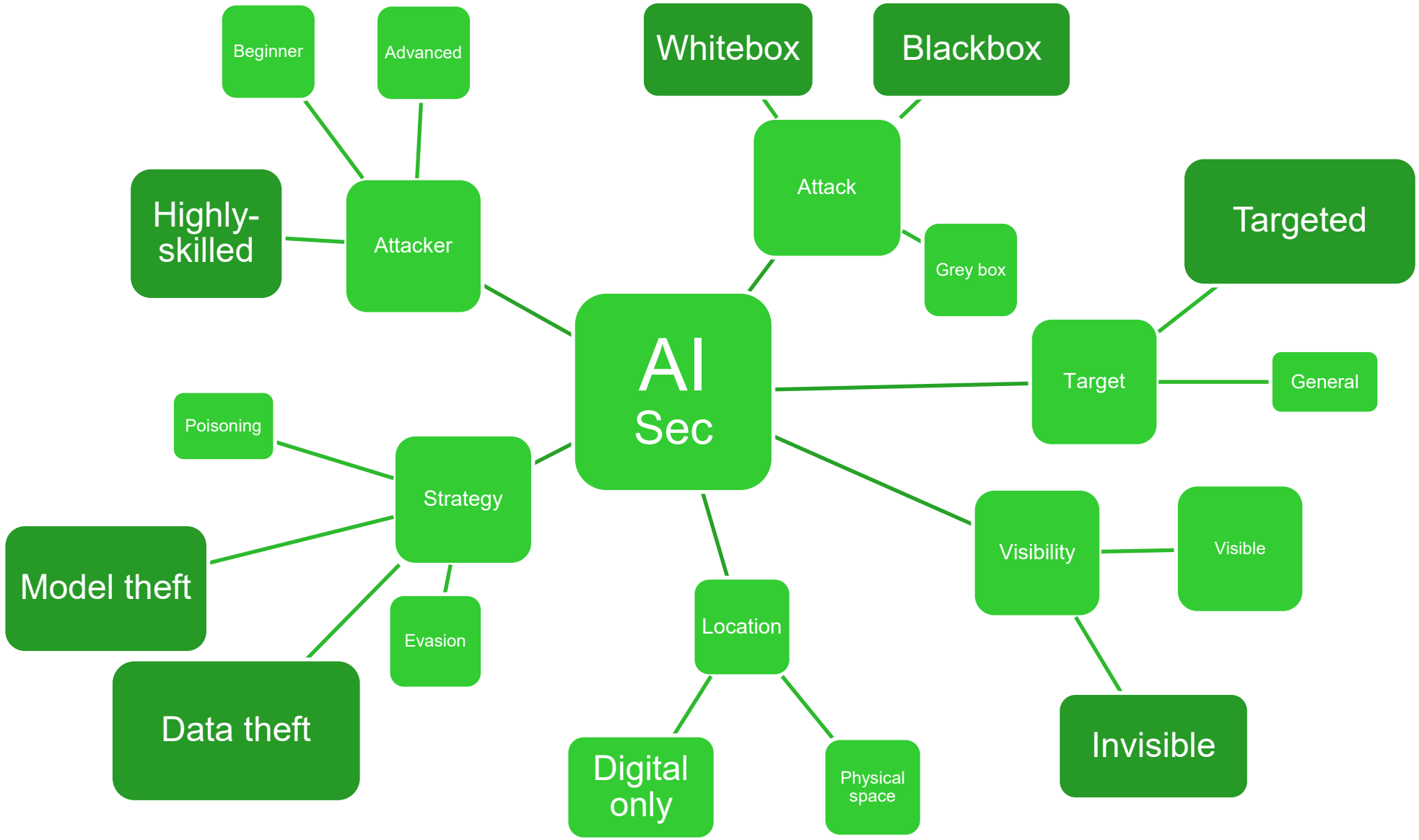
- Model-specific attacks
  - ML05:2023 Model Theft
  - Countermeasures
- Data-specific attacks
  - ML03:2023 Model Inversion Attack → **Discuss:** Why in the name is this “model”-specific?
  - ML04:2023 Membership Inference Attack
- Data-specific countermeasures (in detail)
  - Anonymization, differential privacy, synthetic data generation
  - Federated learning (FL)
  - Homomorphic encryption

Confidentiality in the AI lifecycle



Note: Italic font marks the confidentiality-preserving, AI-specific countermeasures

Privacy attacks

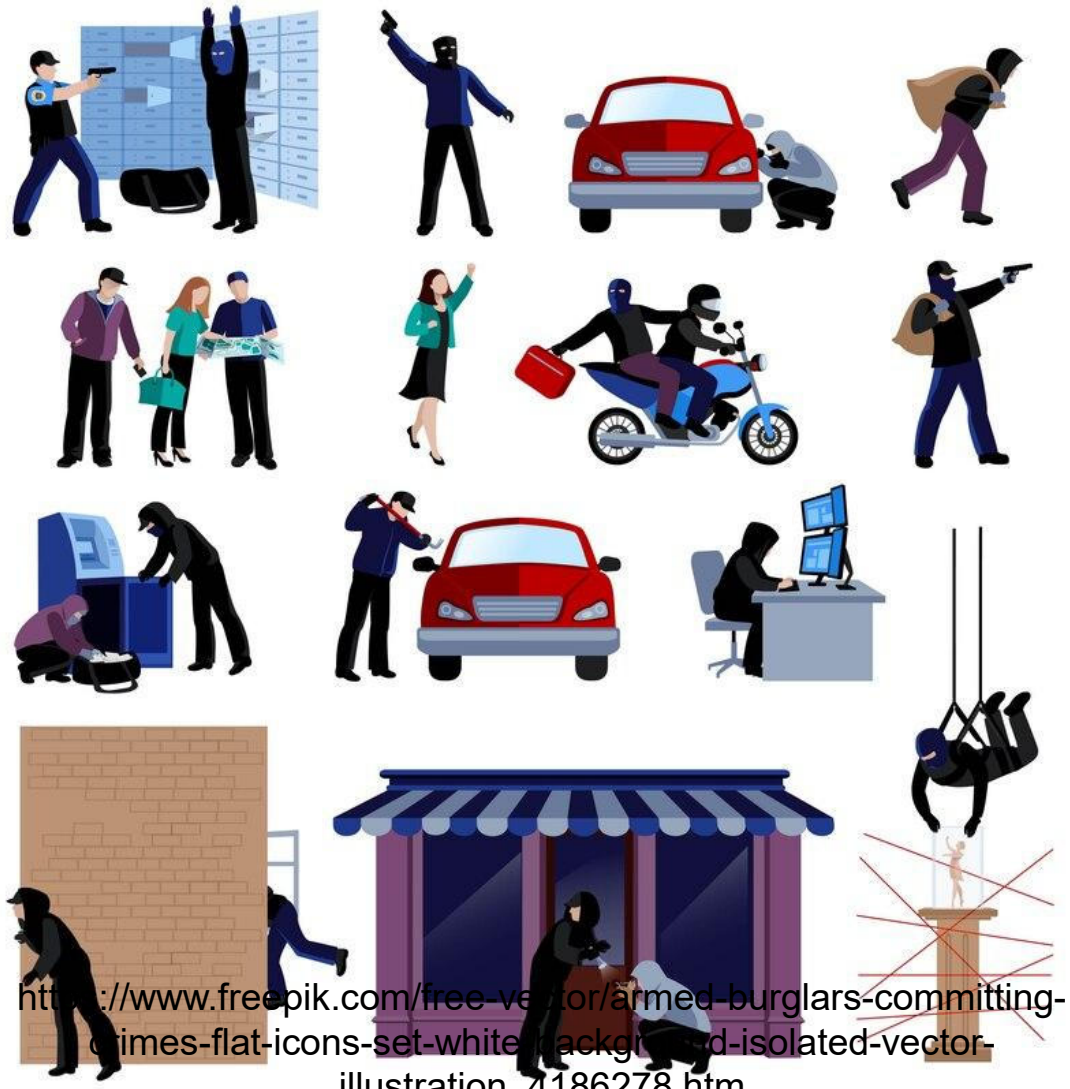


MODEL-SPECIFIC ATTACKS

https://informationmatters.net/data-poisoning-ai/

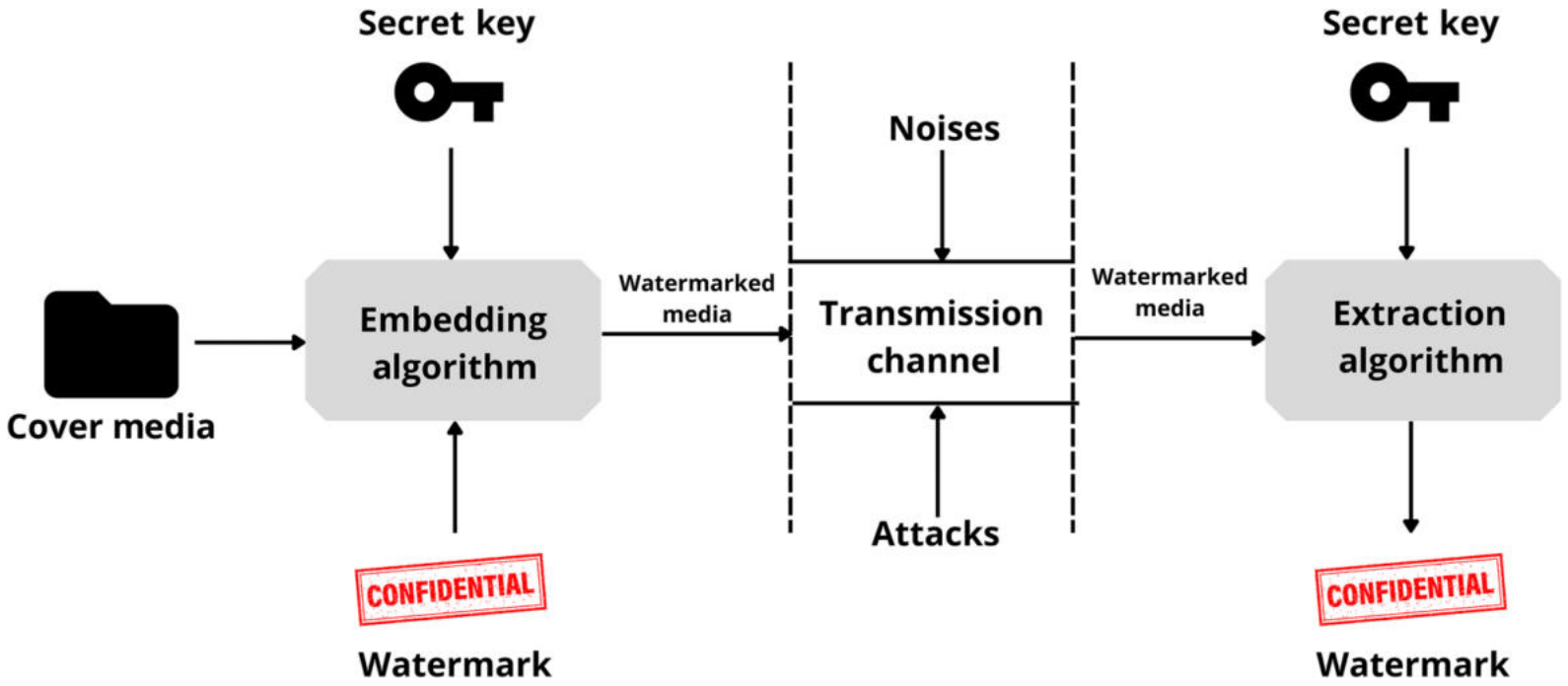
ML05:2023: Model Theft

- **Description:** The attacker gains unauthorized access to the trained model → model duplication
  - Threat sources: API vulnerabilities, insiders
- **Stage:** AI model use/exploitation (maybe training as well)
- **Example** (nothing proven in the wild):
  - Industrial competitor steals model to gain market advantage
- **AI-specific mitigation:**
  - Model watermarking
  - Model obfuscation



http://www.freepik.com/free-vector/armed-burglars-committing-crimes-flat-icons-set-white-background-isolated-vector-illustration_4186278.htm

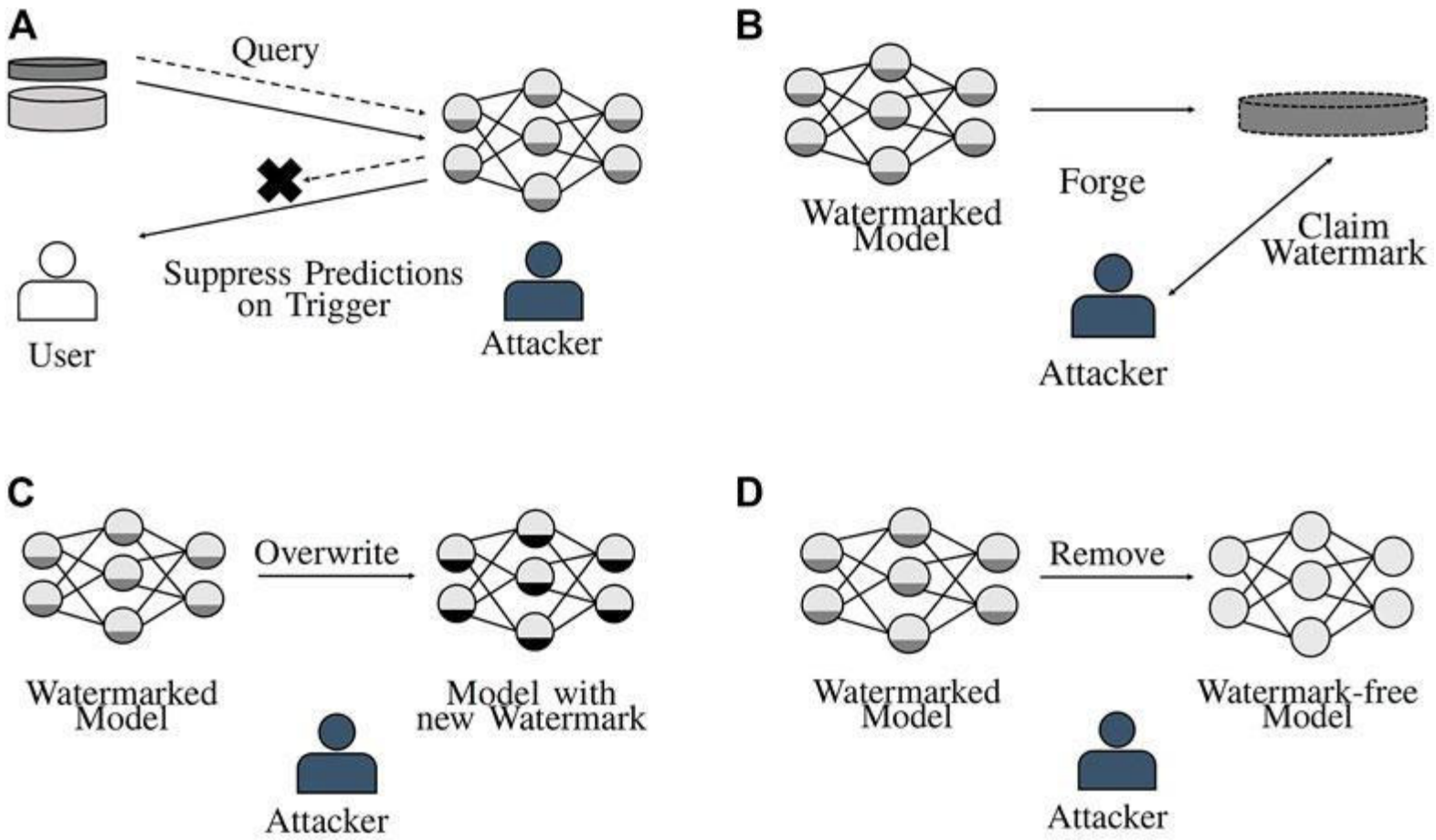
Model watermarking



- **DEF:** A watermark is “a faint design made in some paper during manufacture that is visible when held against the light and typically identifies the maker” (Google dictionary)

- Watermarks can be embedded into ML models → Proof of (ML model) maker
- Types:
  - Traditional → Secret key in content (digital signature?)
  - Model-based → Alter model parameters
  - Learning-based → Certain output for specific input (trigger!)
  - **Discuss:** Learning-based vs backdoor?
- **Discuss:** Watermark removal attacks?

Watermark removal attacks

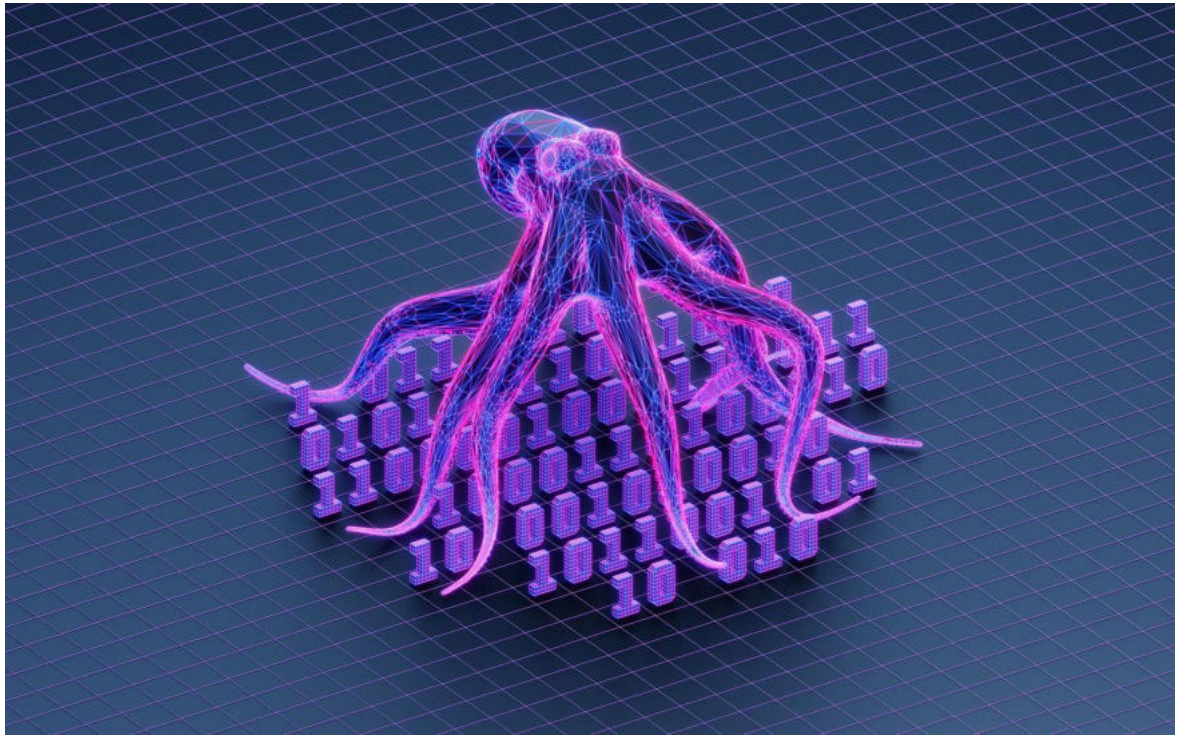


- Watermark attack classes by impact:
  - Detect
  - Suppress
  - Forge
  - Overwrite
  - Remove

- **Discuss:** How to mitigate above attacks?

https://www.frontiersin.org/journals/big-data/articles/10.3389/fdata.2021.729663/full

Model obfuscation



https://www.verimatrix.com/cybersecurity/cybersecurity-insights/the-importance-of-code-obfuscation-and-polymorphism-to-application-security/

- **DEF:** Model obfuscation hides the internal structure of an AI model
  - Model obfuscation is relevant in ML models which can be directly accessed by users → Relevance in white or gray box attacks (!)
  - Similar technique to code (source or binary) obfuscation
  - NOTE: LLMs are self-obfuscated → **Discuss!**
- Obfuscation based on target:
  - File-based (on the serialized model): compression, encryption
  - Model-based: hide model architecture and/or parameters

DATA-SPECIFIC PRIVACY ATTACKS

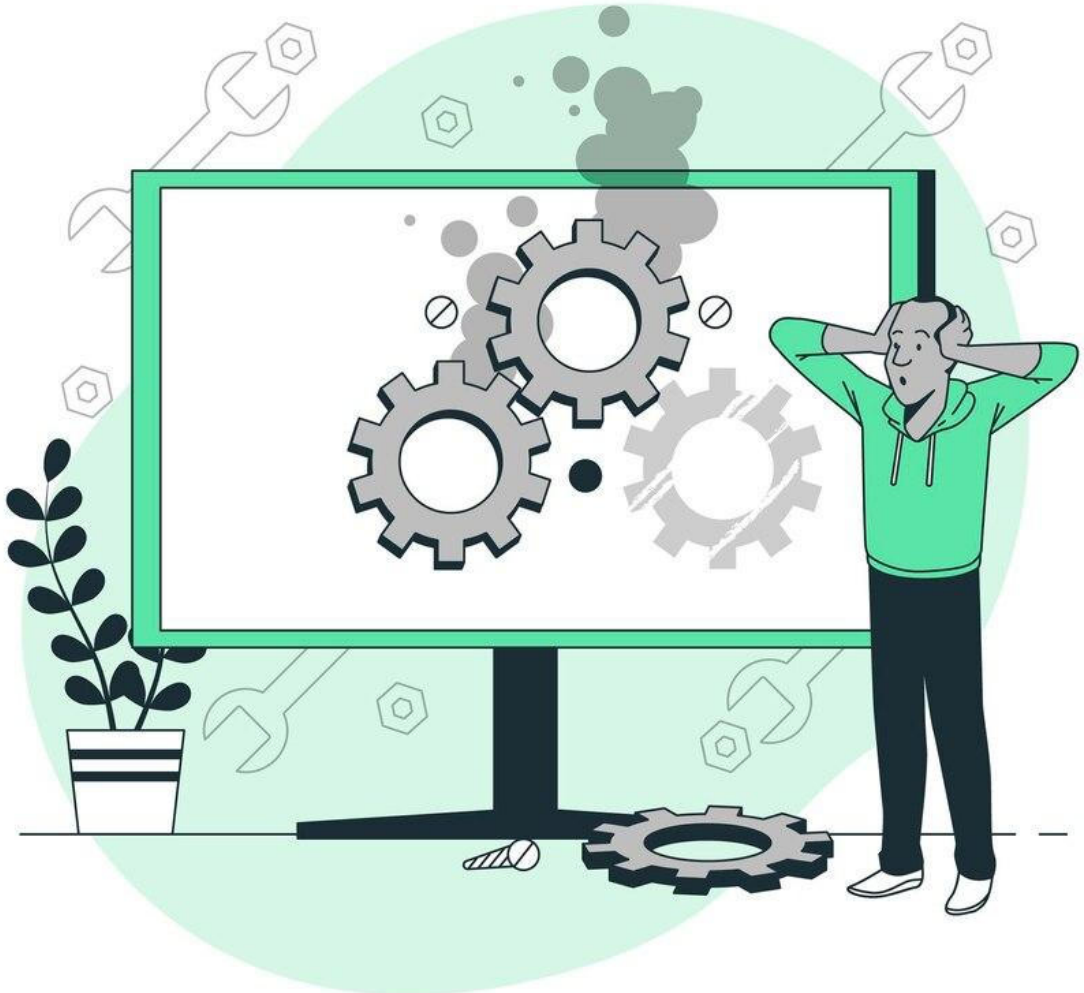
ML04:2023: Membership Inference Attack

- **Description:** The goal is to determine if a specific observation was in the training data or not
  - AI model output (often) differ for data 'seen' compared to data 'not seen' → confidence scores are good indicators
  - Shadow models trained on similar data (!) → **Discuss:** How are these used?
- **Stage:** AI model use/exploitation
- **Examples** (nothing proven in the wild):
  - Healthcare and medical data
- **AI-specific mitigation:**
  - Differential privacy (Strong measure!)
  - Regularization techniques

https://www.freepik.com/free-photo/flat-lay-boss-sticker-mug_9640630.htm

ML03:2023: Model Inversion Attack

- **Description:** The attacker observes model output with the intention to reveal sensitive information contained in training data.
  - Interact with model, observe outputs, infer training data.
  - Overfit models more susceptible. **Discuss:** Why?
- **Stage:** AI model use/exploitation
- **Example:**
  - Infer sensitive information about input data based on output observation
  - **Discuss:** Similar to ML04 Membership inference → What is their relation?
- **AI-specific mitigation:**
  - Security monitoring and anomaly detection (!)
  - Anonymization, differential privacy, synth data
  - Federated learning
  - Input data validation → Very difficult in this scenario
  - Regular retraining
  - Explainable AI and model transparency



https://www.freepik.com/free-vector/computer-troubleshooting-concept-illustration_19184617.htm

Additional confidentiality attacks

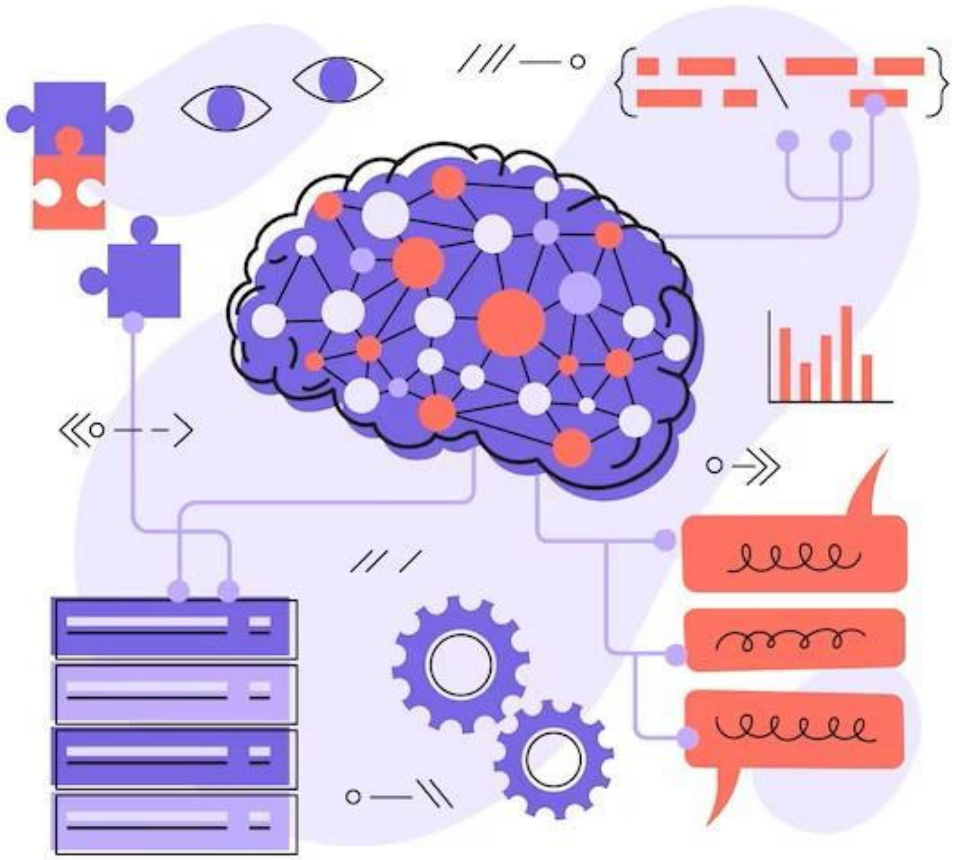
- Additional training phase confidentiality risks:
  - **Algorithm theft** → Allows them to train similar model → **NOTE:** This is only meaningful if training data is also available and model is private (non-public!)
  - **Hyperparameter theft** → Allows them to optimize training of surrogate model → Similarly to algo theft, only meaningful if training data of similar quality is available
- Additional exploitation phase confidentiality risks:
  - **Input data theft** → Monitor what AI system users are submitting to the model → Serious privacy breach possible (!)
  - **Output data theft** → Monitor AI system outputs → NOTE: Only meaningful if inputs are intercepted as well (!)
  - **AI system operations environment theft** → Learn how an operational AI system is built and replicate it for own benefit (!)
- Discuss → Any other ideas of what attackers might exfiltrate/intercept?

COUNTERMEASURES: ANONYMIZATION

Anonymization in a nutshell

- **DEF:** The goal of data anonymization is to perform one-way transformations of sensitive data (the object) which keeps its statistical properties but does not allow attackers to identify data subjects
- Anonymization techniques need to meet two opposite goals:
  - Avoid unwanted leakage of sensitive data (personal or other!), and
  - Avoid data quality loss which would result in catastrophic reduction of ML performance metrics.
- Application domains:
  - Data acquisition and storage at Central Statistical Bureaus (e.g., Központi Statisztikai Hivatal (KSH)) e.g., census data
  - Healthcare data collection, storage and processing (for analytics/research purposes, not individual medical records!)
- **Discuss:** Other application domains?

Naïve data anonymization steps



1. Analyze features in dataset
2. Identify sensitive (personal or otherwise) features e.g., name, address, birth date, religious belief, medical information.
3. Replace sensitive features with random values
   - **Discuss:** How to avoid catastrophic data loss?

- **Discuss:** How to ensure that remaining data values are not usable to deanonymize the data?
- **Discuss:** Any additional challenges?

COUNTERMEASURES: DIFFERENTIAL PRIVACY

Differential privacy in a nutshell

- **DEF:** The goal of differential privacy (DP) is to add minimum noise (ε) to the raw data and thereby obtain anonymous data
  - DP is essentially an anonymization approach (!)
  - Usually applicable in the context of large volumes of input data
- Application domains:
  - Apple: Usage information extracted from iPhone and Mac devices
  - Meta (Facebook): User behavior data analyses in targeted marketing campaigns
  - Alphabet (Google): Historical (vehicle) traffic information collection and sharing → **Discuss:** Waze and Google Maps navigation (Q: What data is needed and what is shown?!)
- Differential privacy types in the function of DP operator execution location (i.e., where do we add minimum noise ε?)
  - **Local:** Noise added at the data acquisition side and before sending to a central data acquisition and storage location e.g., on customer device.
  - **Central:** Noise added in a central location (usually server) where the data is used to train an ML model.
  - **Discuss:** Pros and cons of local vs central DP operator?

DP algorithms

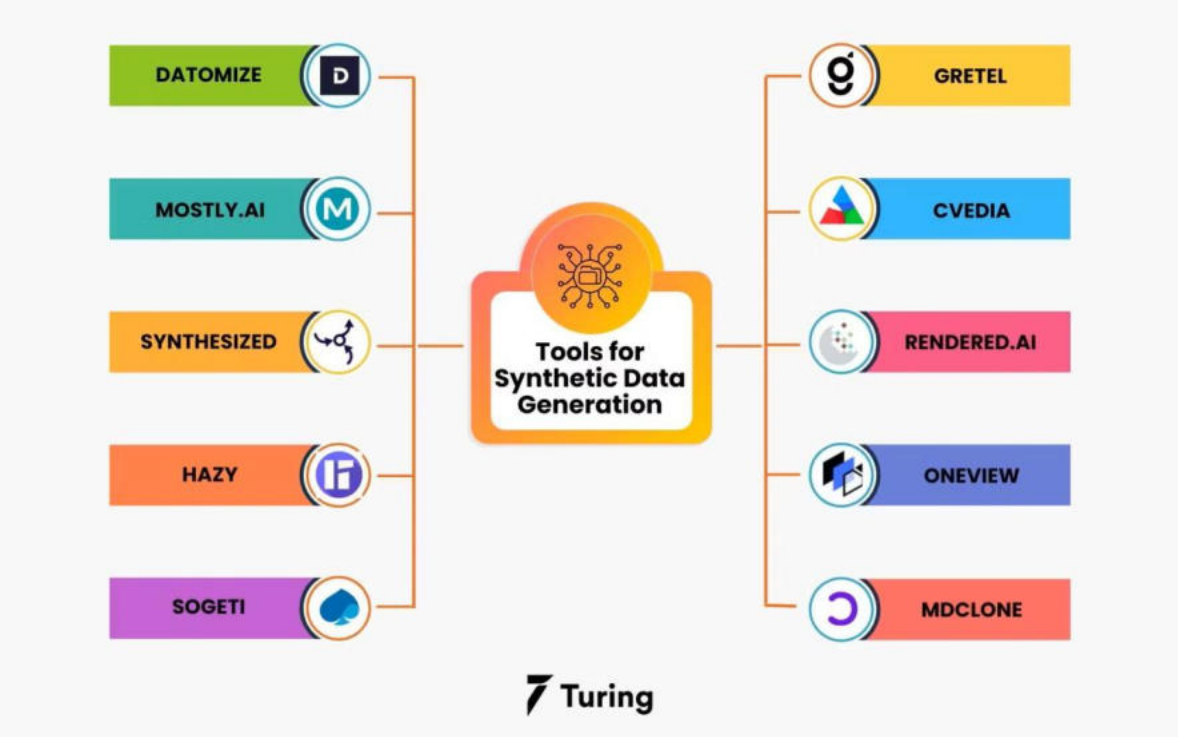
- **Naïve algo:** Add random noise, but do not modify observations above predefined threshold
- **Laplace method:** Add Laplacian noise
- **Exponential mechanism:** Modify data query interface and provide suboptimal answers with certain probability (do not provide TOP(1), but worse result lower down the list of similar answers (!))
  - **Discuss:** Is this applicable in both the ML and DB domains?

- Near & Abuah, Programming differential privacy, https://programming-dp.com/ch9.html
- Task Christine (2012). A Practical Beginners' Guide to Differential Privacy. https://www.youtube.com/watch?v=Gx13lgEudtU

COUNTERMEASURES: SYNTHETIC DATA

Synthetic data generation (SDG)

SDG tools



Turing (2024), “Synthetic Data Generation: Definition, Types, Techniques, and Tools”, https://www.turing.com/kb/synthetic-data-generation-techniques

SDG in brief

- **DEF:** The goal of synthetic data generation is to extend or replace (sensitive) real data
- Broad types of SDG approaches:
  - Statistical distribution-based
  - Model-based: Observe and model behavior → Use model to generate data
- SDG challenges:
  - User acceptance
  - Maintain both statistical similarity and data utility in ML tasks
  - Replicate representative outliers (in anomaly detection tasks)

SDG algorithms

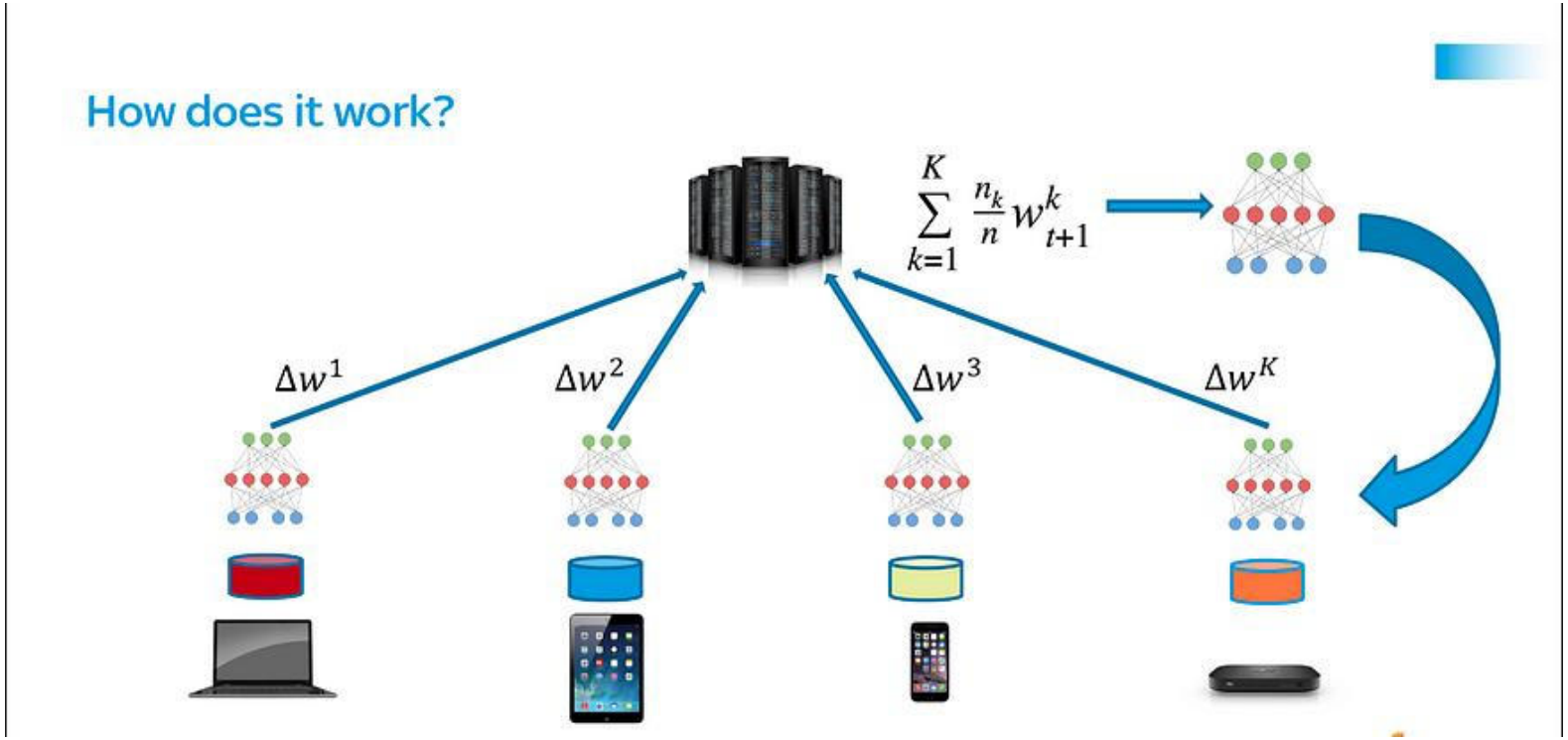
https://www.freepik.com/free-ai-image/digital-art-ai-technology-background_268179471.htm

- **Tabular data:** Gaussian copula (joint distribution-based), Generative Adversarial Networks (GAN)
- **Multimedia:** Diffusion models (very popular!), also GANs, variational autoencoder (VAE)
- **Time-series data:** GAN variants, autoencoders (?)
- **Discuss:** Evaluation of SDG data and algorithms in the function of data type?
- References:
  - https://medium.com/@ashish28082002.ak/tabular-synthetic-data-generator-using-conditional-gans-d98fcd974148
  - https://medium.com/@sivavimelrajhen/a-basic-introduction-to-image-generation-methods-25719fdea31e

COUNTERMEASURES: FEDERATED LEARNING

Federated learning in a nutshell

1. Do not collect sensitive data in a central location at all
2. Training partial models on the premises and devices of participants
3. Integrate partial models in a central location (which does not see training data!)



FL challenges

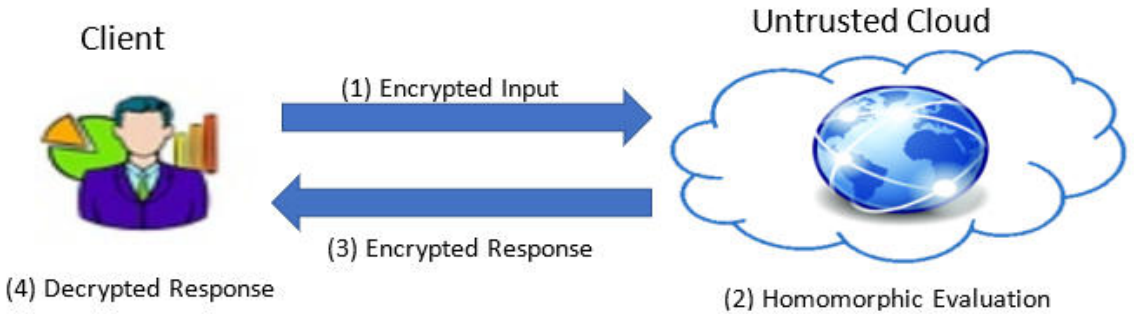
Image by macrovector on Freepik

- FL challenges inherited from traditional distributed systems:
  - Lack of CPU/GPU capacity
  - Communication bottlenecks and other problems (loss of packets, delays, etc.)
- FL-specific challenges
  - Poor performance (of central model)
  - The training algorithm might not converge at all
  - Data inversion at and of the central model
  - Model poisoning by malicious FL actors
  - **Discuss:** Any other FL-specific challenges?

Countermeasures: Homomorphic encryption

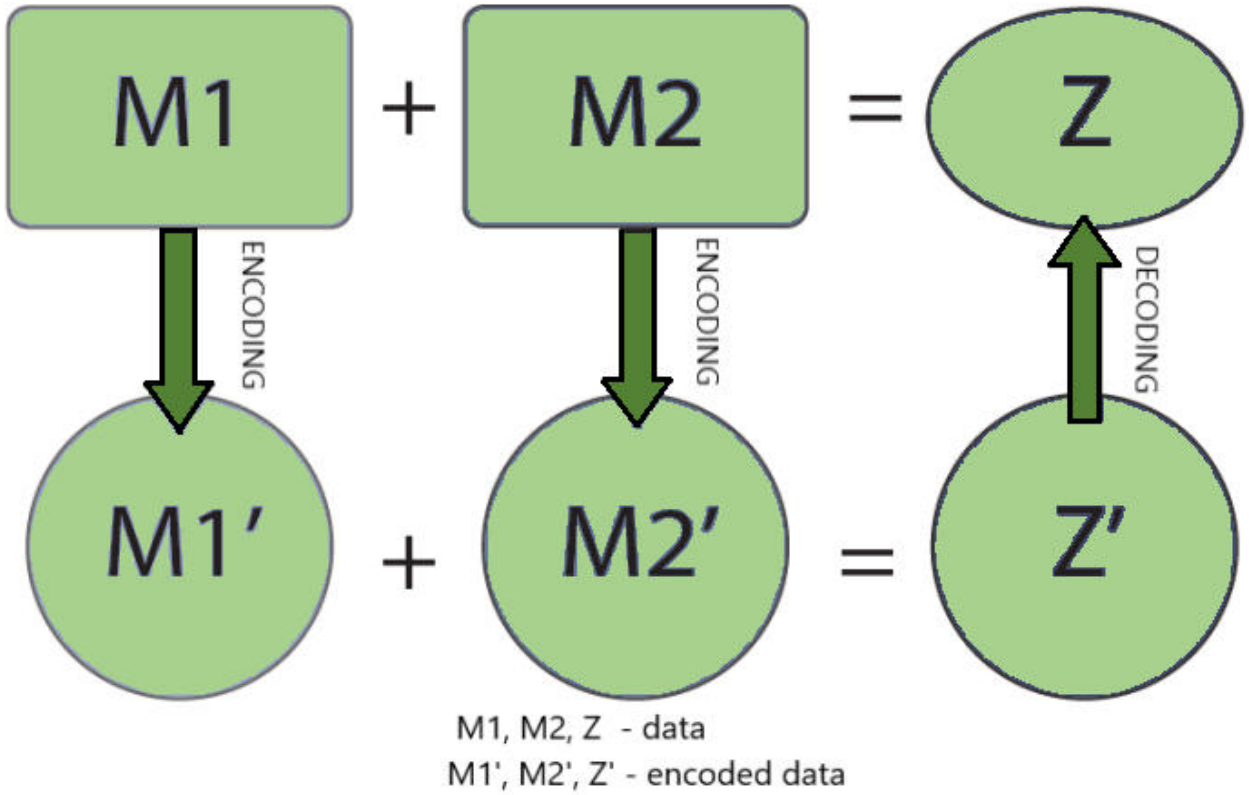
Homomorphic encryption in a nutshell

- **DEF:** The goal of homomorphic algorithms is to allow certain operators to work on encrypted data
  - Partial or complete algorithms → Either or both additive and/or multiplicative
- Meaning with Greek origin
  - Homo – the same (not „human”!)
  - Morphe – form or shape
- Data stage: In use (not ’in transit’ and not ‘at rest’)



https://www.intel.com/content/www/us/en/developer/articles/technical/homomorphic-encryption/accelerating-homomorphic-encryption-for-fpga.html

Operation



“Partially Homomorphic Encryption with RSA in Python From Scratch”, RSA Conference 2023, https://www.youtube.com/watch?v=PzKch8UQAmQ

Application domains

- Sensitive data analytics in the computing cloud
- Multi-party calculus
- Acquisition and analysis of health data
- Image recognition on encrypted data
- Digital-only evidence in the context of digital forensics
- Data protection on mobile devices

Image by rawpixel.com on Freepik

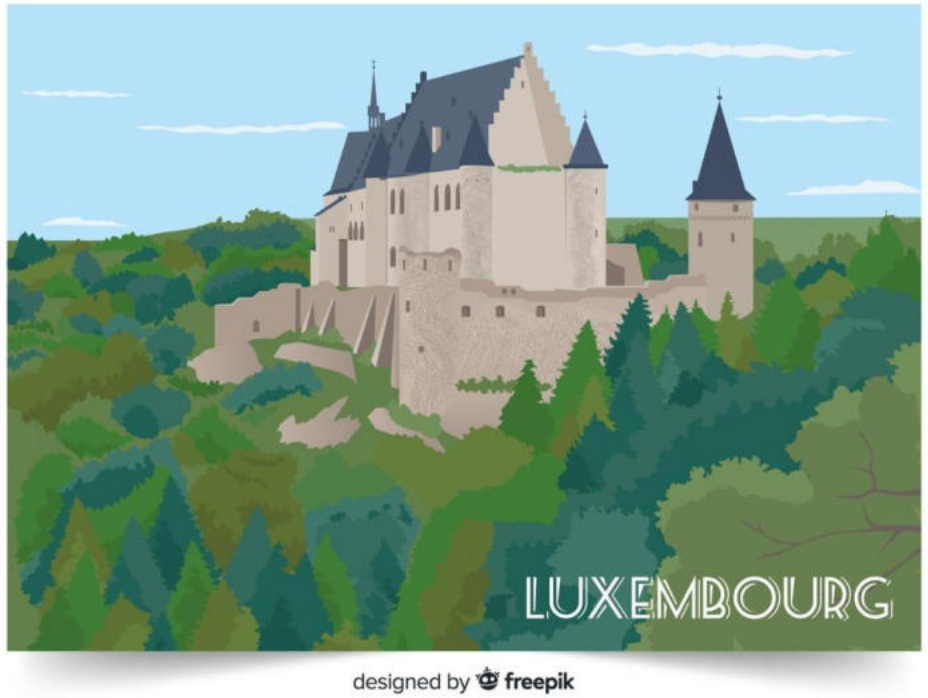
Partial: RSA

- **Authors:** Ronald Rivest, Adi Shamir, and Leonard Adleman (1978)
- **Trivia:** Authors defined private homomorphism as a concept
- **Operator supported:** Multiplication
- **Mathematical foundation:** Large prime factorization
- **Additional sources:**
  - “Machine Learning over Encrypted Data with Fully Homomorphic Encryption”, https://www.youtube.com/watch?v=8tWAxUgO2V0
- **Practicals:** CyberChef

Image by vectorjuice on Freepik

Complete: Gentry

- **Author(s):** Craig Gentry, PhD thesis (2009)
- **Relevance:** First complete homomorphic algorithm
- **Supported operators:** Multiplication, addition
- **Math foundation:** Lattice-based
- **Sources:**
  - Craig Gentry, “How would you explain homomorphic encryption?”, https://www.youtube.com/watch?v=pXb39wj5ShI
  - Chalk Talk, “Lattice-based cryptography: The tricky math of dots”, https://www.youtube.com/watch?v=QDdOoYdb748



SUMMARY & NEXT STEPS

Summary and next steps

- **Model-specific** attacks
  - ML05:2023 Model Theft
- Confidentiality attacks targeting the **training data**:
  - ML03:2023 Model Inversion Attack → We discussed the lack of naming focus (!)
  - ML04:2023 Membership Inference Attack
- Additional targets of confidentiality attacks (!)
- Confidentiality (and privacy) enhancing **countermeasures**:
  - Model-focused: watermarking, obfuscation
  - Data-focused: anonymization, differential privacy, synthetic data generation, federated learning, homomorphic encryption

Additional references – 1

- Franziska Boenisch (2021), “A Systematic Review on Model Watermarking for Neural Networks”, https://doi.org/10.3389/fdata.2021.729663
- Galloni, A., Lendák, I., & Horváth, T. (2020, October). A novel evaluation metric for synthetic data generation. In *International Conference on Intelligent Data Engineering and Automated Learning* (pp. 25-34). Cham: Springer International Publishing.
- Kendrick Boyd (2022), “Generate Synthetic Time-series Data with Open-source Tools”, https://www.kdnuggets.com/2022/06/generate-synthetic-timeseries-data-opensource-tools.html
- Thiwanka Chameera Jayasiri (2023), “Watermarking Machine Learning Models: A Pathway to Model Verification and Authorship Assertion”, https://medium.com/@thiwankajayasiri/watermarking-machine-learning-models-a-pathway-to-model-verification-and-authorship-assertion-71e3f3d10bc6
- John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, Tom Goldstein (2023), “A Watermark for Large Language Models”, https://arxiv.org/abs/2301.10226
- NIST (2024), “Adversarial Machine Learning: A Taxonomy and Terminology of Attacks and Mitigations”, https://doi.org/10.6028/NIST.AI.100-2e2023

Additional references – 2

- Rashidi, H.H., Albahra, S., Rubin, B.P. et al. A novel and fully automated platform for synthetic tabular data generation and validation. Sci Rep 14, 23312 (2024). https://doi.org/10.1038/s41598-024-73608-0
- Rod Trent (2023), “Must Learn AI Security Part 6: Model Inversion Attacks Against AI”, https://rodtrent.substack.com/p/must-learn-ai-security-part-6-model
- Rod Trent (2023), “Must Learn AI Security Part 7: Membership Inference Attacks Against AI”, https://rodtrent.substack.com/p/must-learn-ai-security-part-7-membership
- Rod Trent (2023), “Must Learn AI Security Part 8: Model Stealing Attacks Against AI”, https://rodtrent.substack.com/p/must-learn-ai-security-part-8-model
- Rod Trent (2023), “Must Learn AI Security Part 21: Watermark Removal Attacks Against AI”, https://rodtrent.substack.com/p/must-learn-ai-security-part-21-watermark
- Mingyi Zhou at al (2024), “ModelObfuscator: Obfuscating Model Information to Protect, https://arxiv.org/pdf/2306.06112

Thank you for your attention!

Eötvös Loránd University (ELTE)
Faculty of Informatics (IK)
Pázmány Péter sétány 1/c
1117 Budapest, Hungary

AVAILABILITY ATTACKS

Lecture #5 in Introduction to Data Security

Imre Lendák, PhD, GICSP

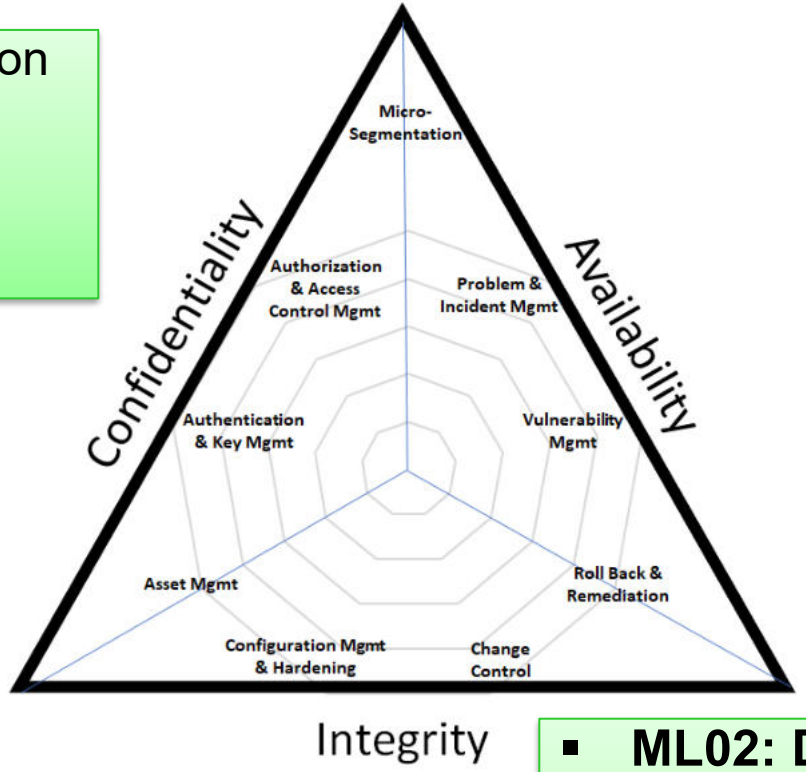
2026
Budapest, Hungary

Presentation outline

- Introduction & CIA triad
- Sponge attacks

Attack types in the CIA triad…

- ML03:2023 Model Inversion
- ML04:2023 Membership Inference
- ML05:2023 Model Theft

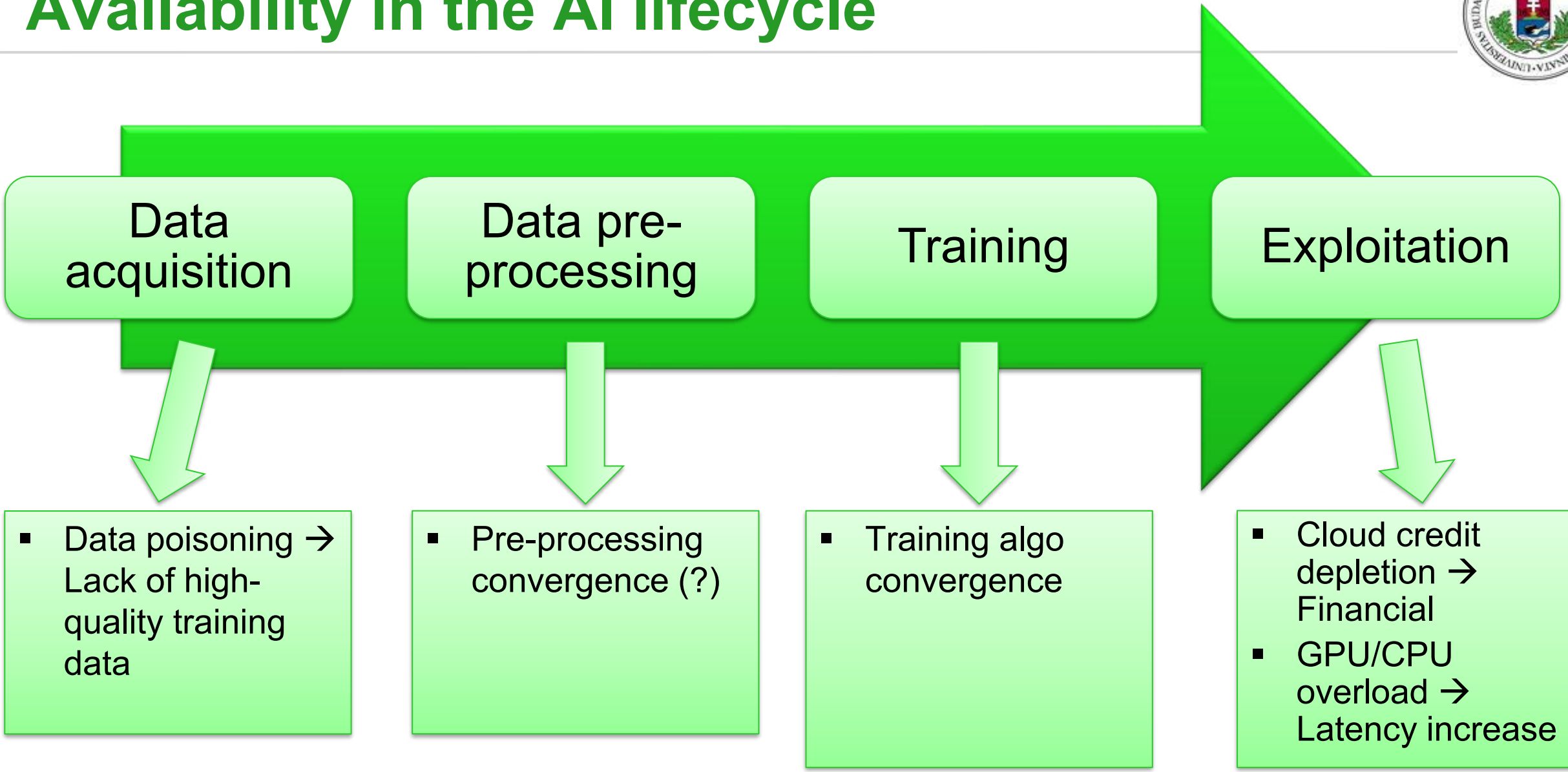


- ML09: Output integrity (?)
- **ML10: Model poisoning**

- **Discuss:** What about **ML01**??? Is it really an integrity attack?

- **ML02: Data poisoning**
- **ML06: AI supply chain**
- ML07: Transfer learning
- ML08: Model skewing
- ML09: Output integrity (?)
- **ML10: Model poisoning**

Availability in the AI lifecycle



SPONGE ATTACKS

Sponge attacks

- The goal of the sponge attack is to confuse, distract or overwhelm the AI → The AI absorbs irrelevant inputs like a sponge (!)
- Adversarial samples → Malicious inputs crafted for sponge attacks → Cause (significantly) more activations in neural networks
  - **Discuss:** White & black box sponge attacks!
- Example: The Tay chatbot (2016) can be (remotely) considered a sponge attack → **Discuss:** Why?
- Specific defenses: rate limiting, adversarial training, AI system performance monitoring (→ detect service degradation)

https://en.wikipedia.org/wiki/SpongeBob_SquarePants_%28character%29

Conclusion

- Introduction & CIA triad
- Availability in
  - Data acquisition
  - Data pre-processing
  - Training
  - Exploitation
- Sponge attacks

References

- Rod Trent (2023), “Must Learn AI Security Part 25: Sponge Attacks Against AI”, https://rodtrent.substack.com/p/must-learn-ai-security-part-25-sponge

Thank you for your attention!

Eötvös Loránd University (ELTE)
Faculty of Informatics (IK)
Pázmány Péter sétány 1/c
1117 Budapest, Hungary

DEFENDING AI SYSTEMS

Lecture #6 in Introduction to Data Security

Imre Lendák, PhD, GICSP

2026
Budapest, Hungary

Presentation outline

- Monitoring in AI ops
- Robust by design
- Defenses vs attack types

MONITORING THE AI SYSTEM

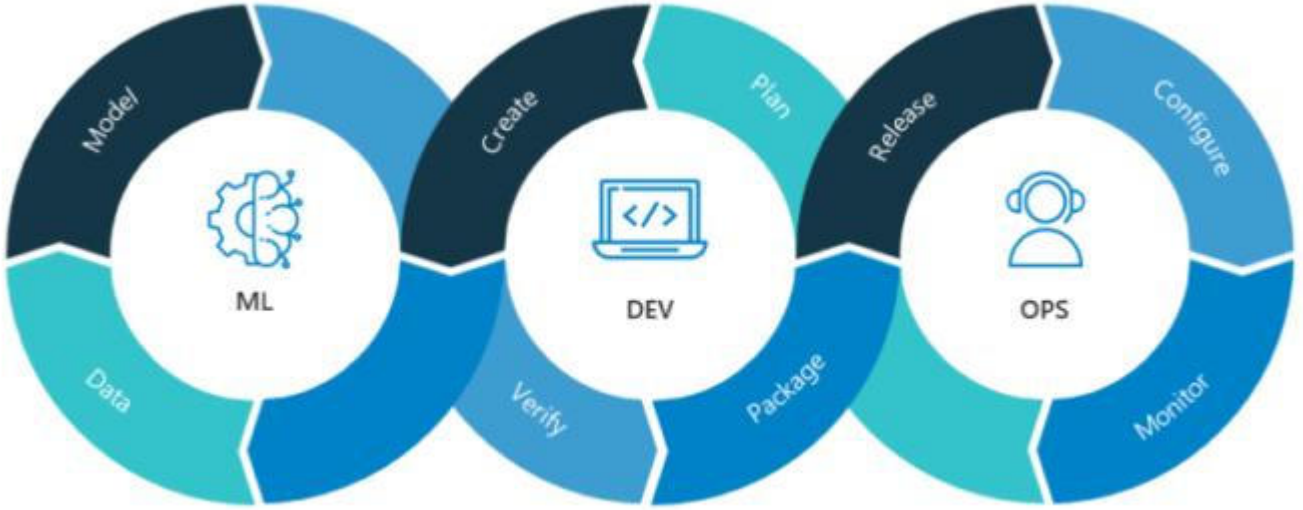
AI training monitoring

- Training-specific key performance indicators (KPI) to be measured:
  - Learning rate & convergence
  - Precision on training and validation data (over time)
  - Training data batch influence on model training / performance (if possible)
  - Regularization results → Detailed discussion in a few slides…
  - Ensemble decision process evaluation → **Discuss:** What does this mean?
  - **Discuss:** What else? How does this influence re-training decisions? Change detection?

https://www.tibco.com/glossary/what-is-data-validation

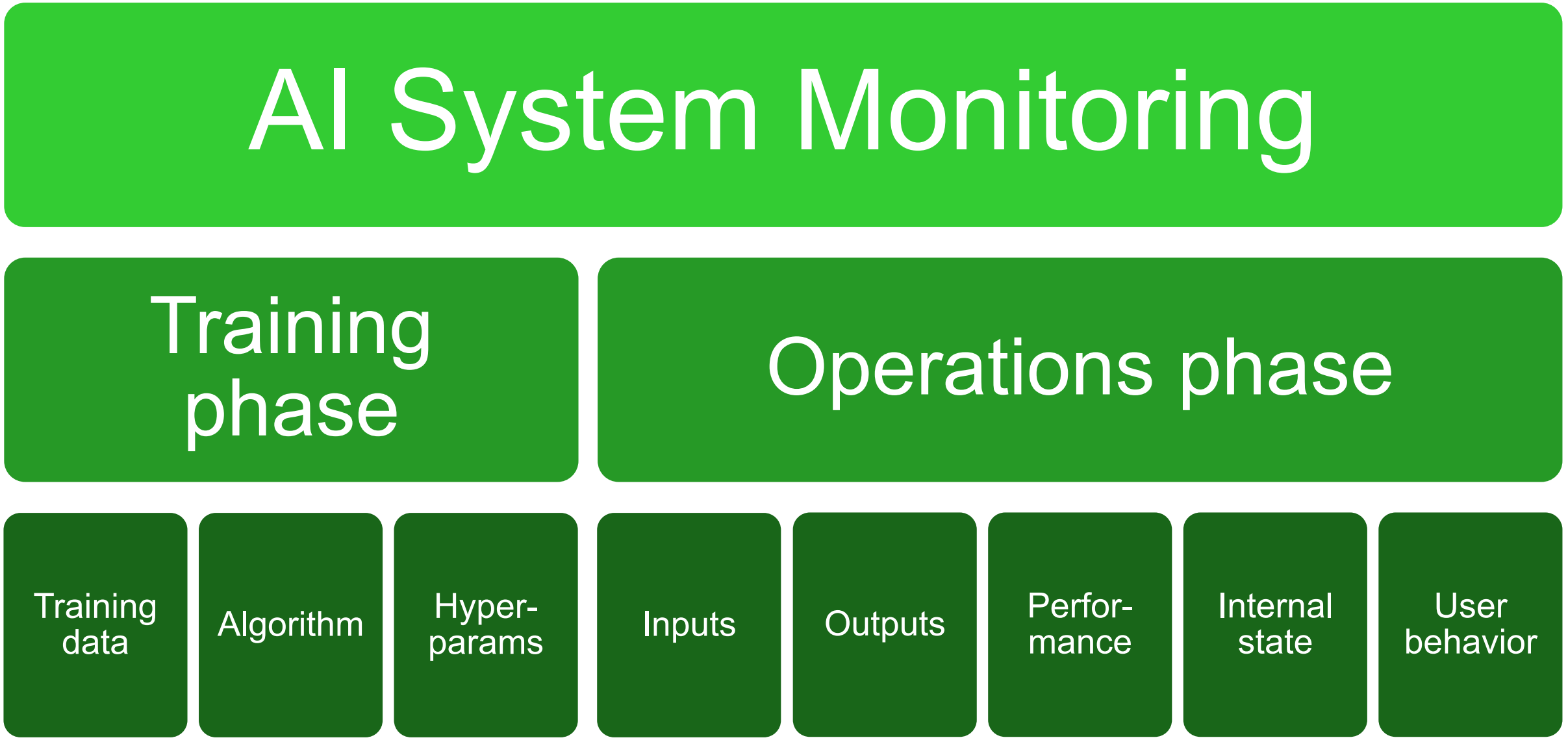
MLops/Alops monitoring

- Analyze input data and filter out (known) malicious samples → Might involve lightweight (ML-based) anomaly detection or expert knowledge (manual labor!)
  - Image analysis → Known (invisible) noise
  - Malware/spam analysis → Detect novel tricks in hiding malicious content
  - LLMs → Malicious prompt detection based on predefined or learned filters
  - Time-series analysis → Rate, statistical variance, change? → **Discuss:** What else?

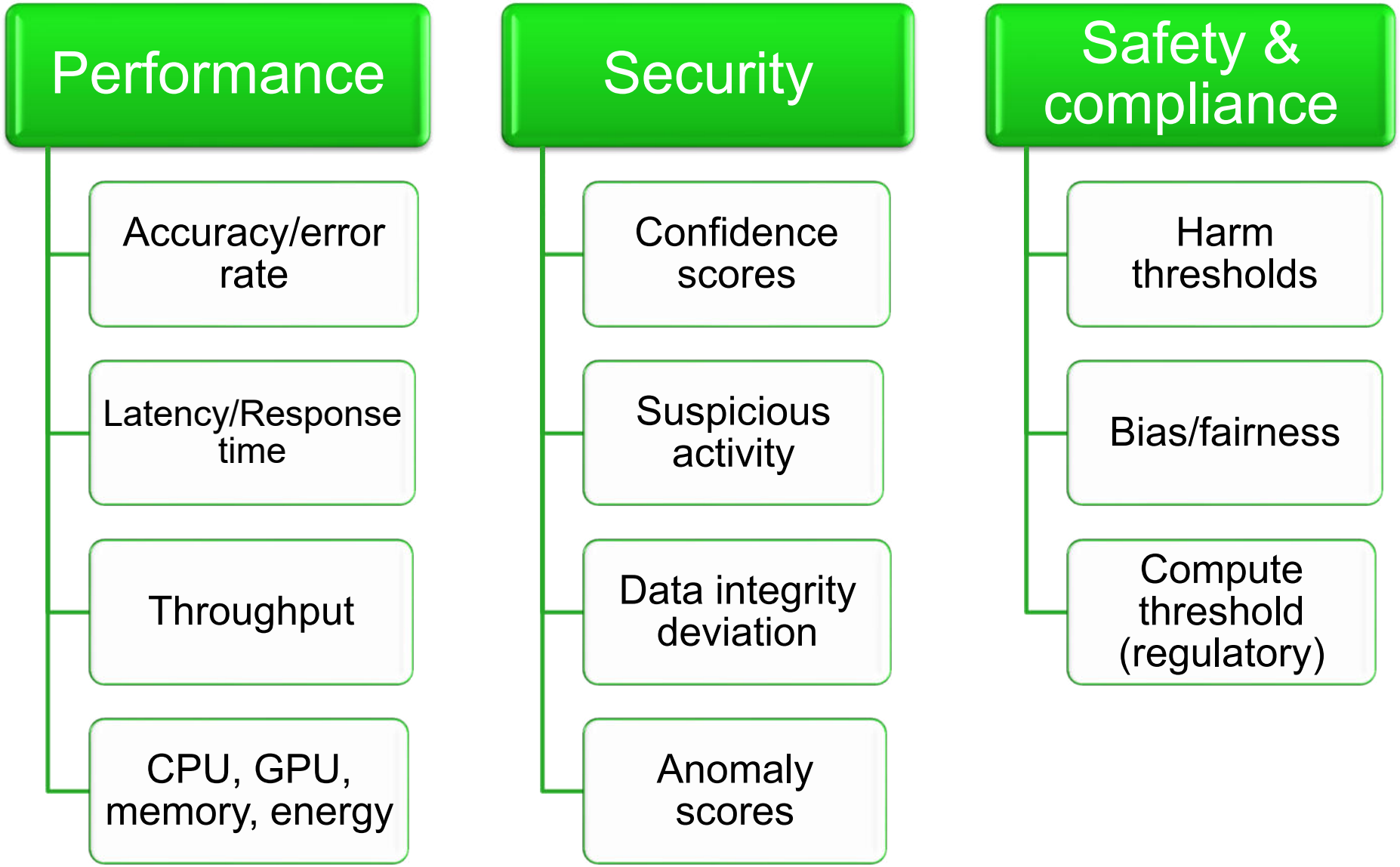


https://blogs.nvidia.com/blog/what-is-mlops/

AI System Monitoring



AI monitoring-based cut-off thresholds



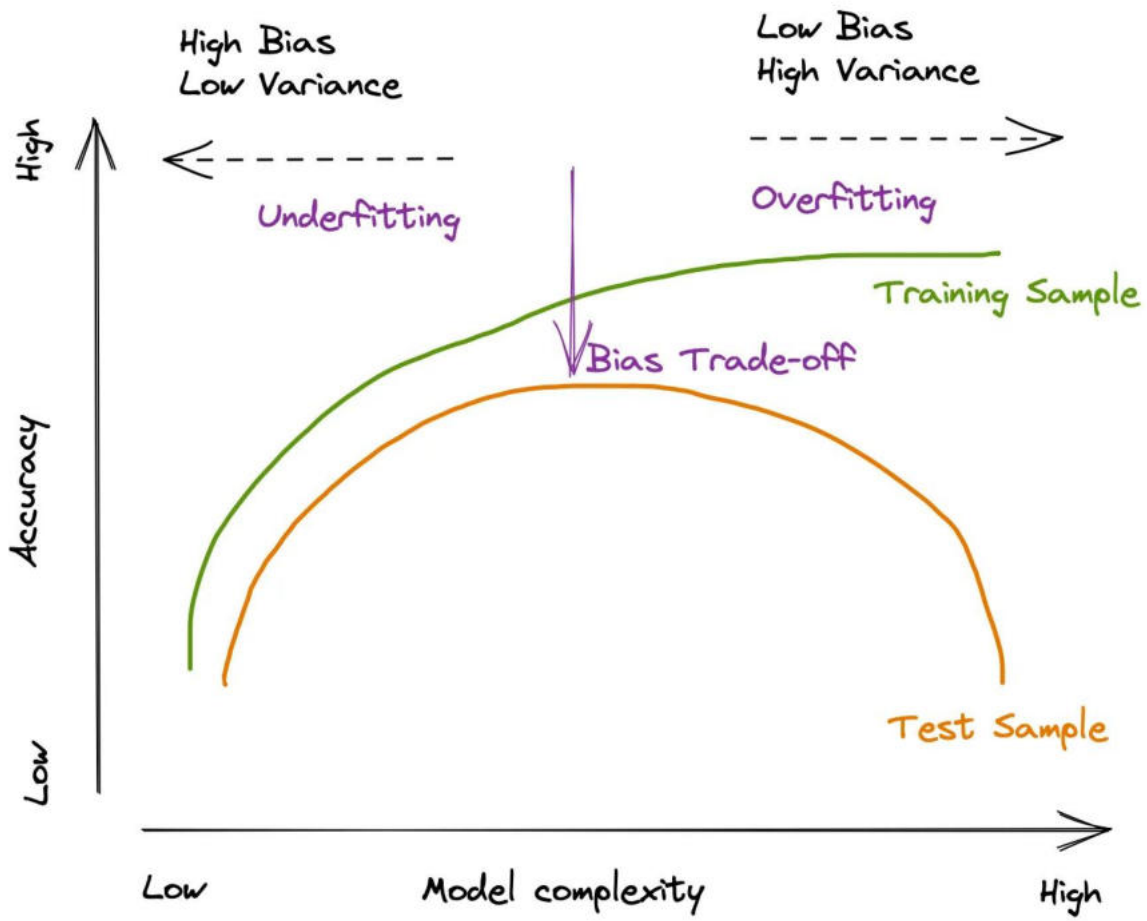
Robust by design vs Robust by training

AI ops infrastructure

- Rate limiting
  - Number of (user) inputs accepted in unit of time
  - Number of cloud credits used in unit of time
  - Energy (CPU/GPU) use per unit of time
- Latency monitoring
  - Input → Output roundtrip length
  - Time spent in each stage (input processing, inference, output filtering)
- User monitoring (e.g. high-cost inputs)
  - Detect suspicious / malicious users based on input to output cost analysis

https://www.freepik.com/free-ai-image/cyber-security-concept-digital-art_236289599.htm

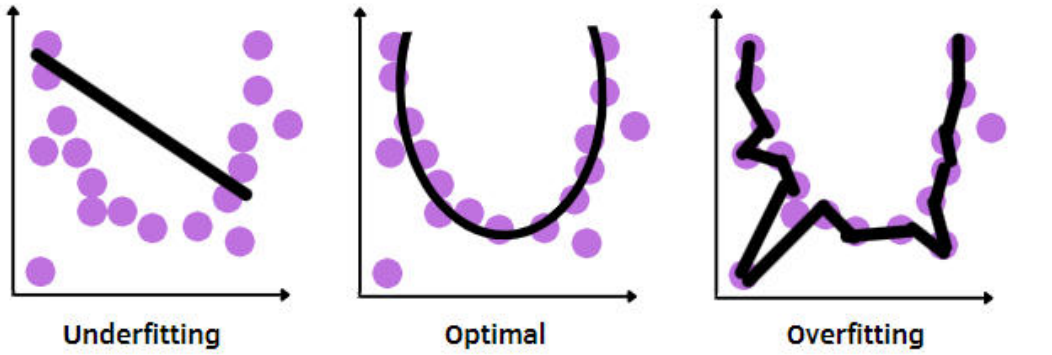
Robust models



https://medium.com/@slavadubrov/understanding-machine-learning-robustness-why-it-matters-and-how-it-affects-your-models-5e2cb5838dab

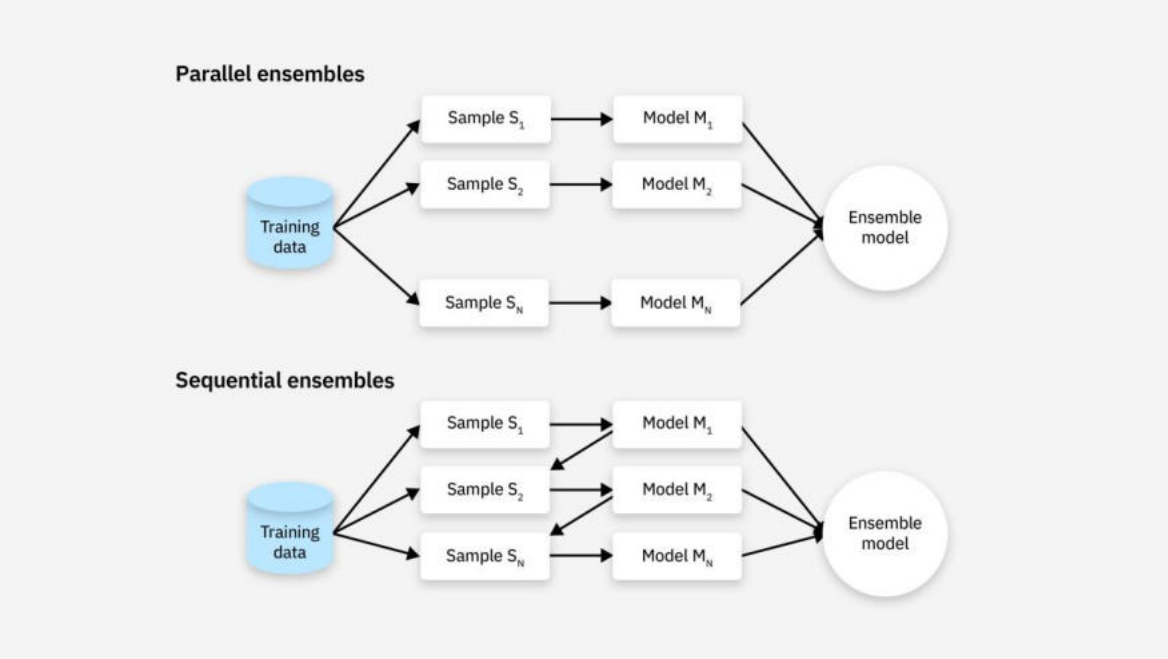
- **DEF:** Model robustness is the ability of the ML model to maintain performance in adversarial conditions
  - Noisy data, distribution shifts (i.e., change), adversarial attacks
- Robust modeling approaches:
  - Augment and preprocess training data
  - Model regularization → Constrain model training, avoid overfit → **Discuss:** Why is this relevant?
  - Ensemble learning with model diversity → Train and apply multiple, diverse models
  - Transfer learning → Fine tune pretrained model
  - Domain adaptation → Adapt to distribution shifts
  - Interpretability and explainability → Specific models → Increased trust

Regularization



- **DEF:** Underfitting means that the model does not capture the relation between inputs and outputs properly → High error rate
- **DEF:** Overfitting means that the model matches the training data too closely, but fails to make correct decisions on new data
- **DEF:** Regularization reduces overfitting by encouraging the model to prioritize simpler hypotheses → See Optimal figure on left (!)
  - Select proper model → Train → Regularize
  - Constrains model complexity e.g., early stopping, pruning (tree-based algos), dropout → **Discuss!**
  - Trade training accuracy for generalizability → Ability to generate correct output on unseen data
  - Penalizes larger weights (in NNs) → **Discuss:** Why?
- AI security relevance:
  - Data and model backdoor removal
  - Increased resistance to training data manipulation e.g., label flipping

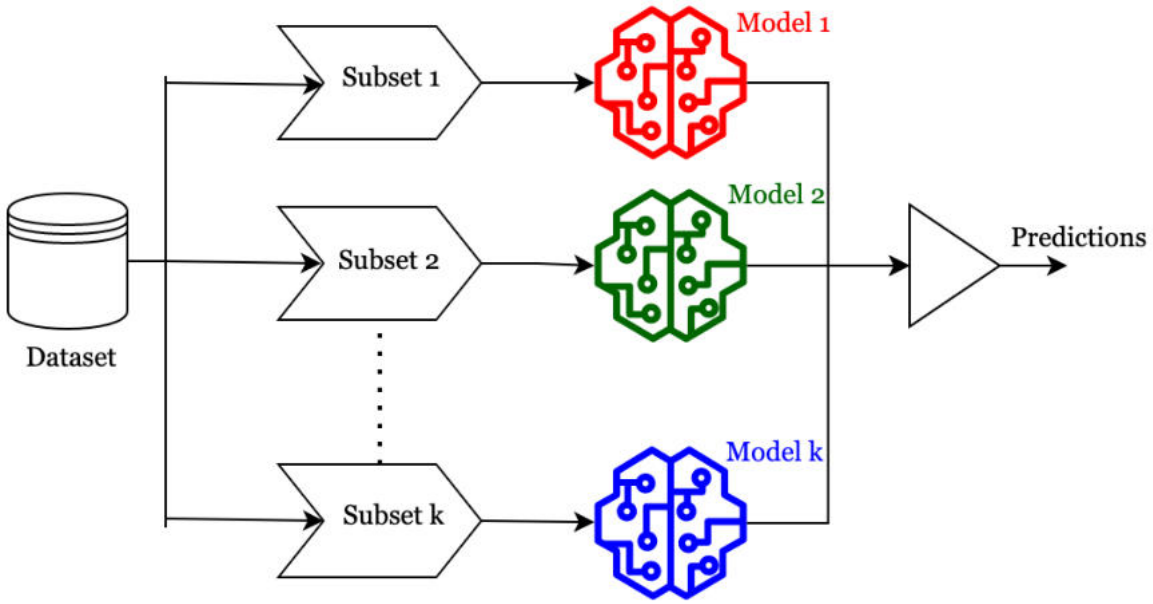
Ensemble learning (1/2)



https://www.ibm.com/think/topics/ensemble-learning

- **DEF:** Ensemble learning aggregates two or more (base) models to improve performance (over a single model)
  - Ideally different model types and architectures
  - Different subsets of training data
  - Models used in operations participate in the vote
  - Models can be switched in and out → **Discuss:** Why & based on what?
- Trivia: Member models are called ‘base estimators’ or ‘baseline models’

Ensemble learning (2/2)



- Ensemble learning combines the outputs of multiple models to obtain improved model performance → Use the collective 'intelligence' of multiple models
- Ensemble types:
  - **Bagging:** Parallel models, majority voting
  - **Boosting:** Sequential models, error correction
  - **Stacking:** Different models' outputs to train a meta learner model
  - **Cascade:** Sequential, more complex models, (maybe) human at the end of the cascade
- Use in AI security: Ensemble member models highly diverse decisions → Possible adversarial sample
  - Detection of adversarial samples e.g., triggering backdoors, copy-move attack → **Discuss:** How?

https://intuitivetutorial.com/2023/05/12/ensemble-models-in-machine-learning/

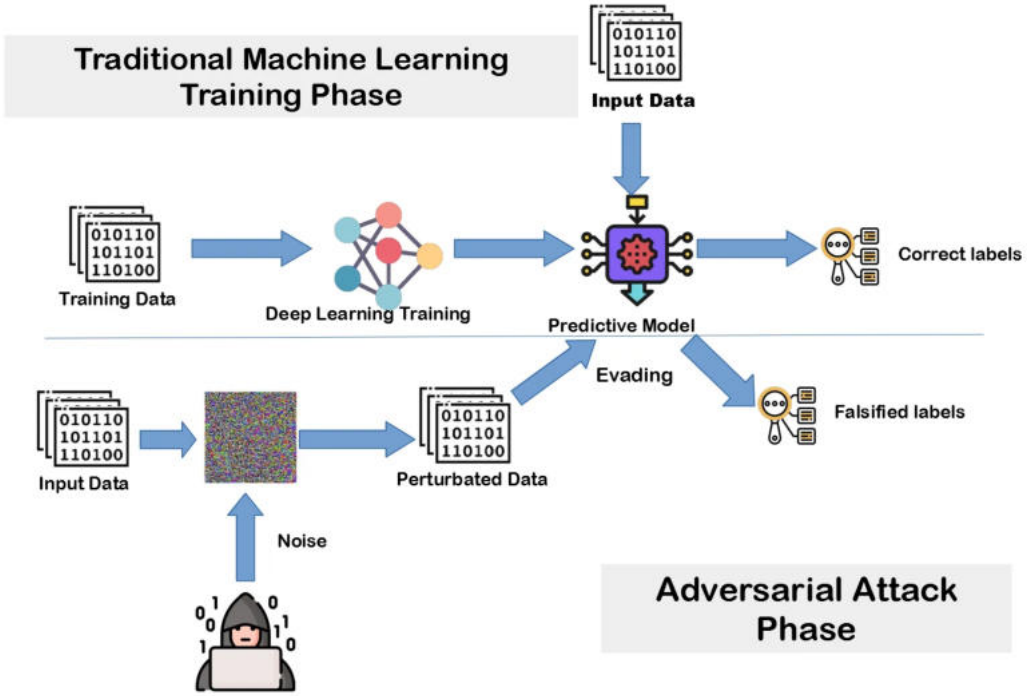
Adversarial training

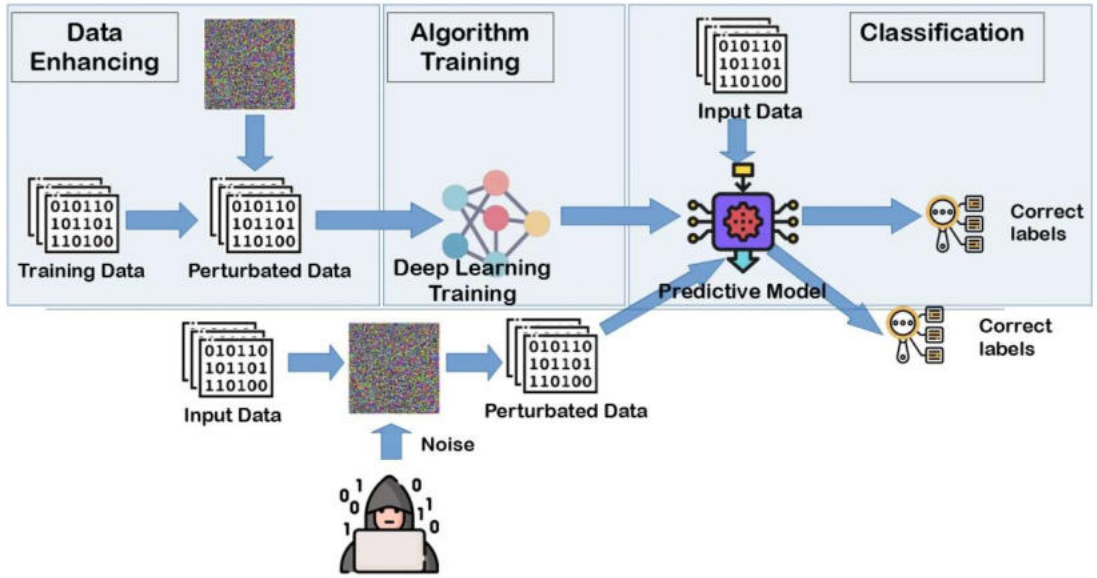
Training with no adversarial data

- ML model susceptive to wrong decisions if adversarial input data

Training with adversarial data

- Collect or synthesize adversarial samples
- Train with augmented data





Defenses vs attack types

Counter measure types

- Data-focused:
  - Training data validation
  - Adversarial training
- Model-focused:
  - Model IO and performance monitoring → Model re-training
  - Use of robust models and robustness testing → Test model sensitivity to specific training data changes
  - Use of model ensembles → Measure and compare performance
  - Model regularization → Expected higher resilience to malicious inputs
- Note: AI-generated figure on the right (!)

https://www.freepik.com/free-ai-image/cyber-security-concept-digital-art_236289599.htm

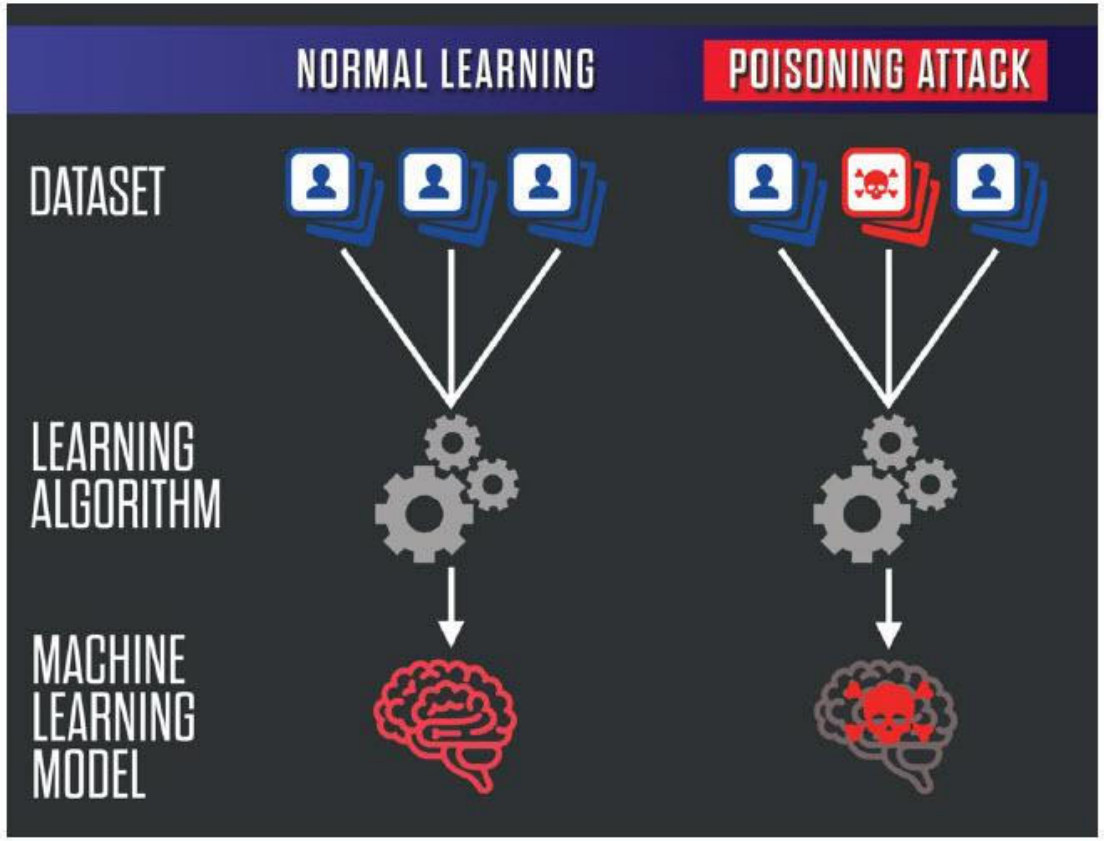
Defenses against input data manipulation

- Adversarial training
- Use of robust models
- Input data anomaly detection and filtering
- Model performance monitoring

- Discuss → Anything else?

https://www.freepik.com/free-photo/ai-cybersecurity-virus-protection-machine-learning_17850530.htm

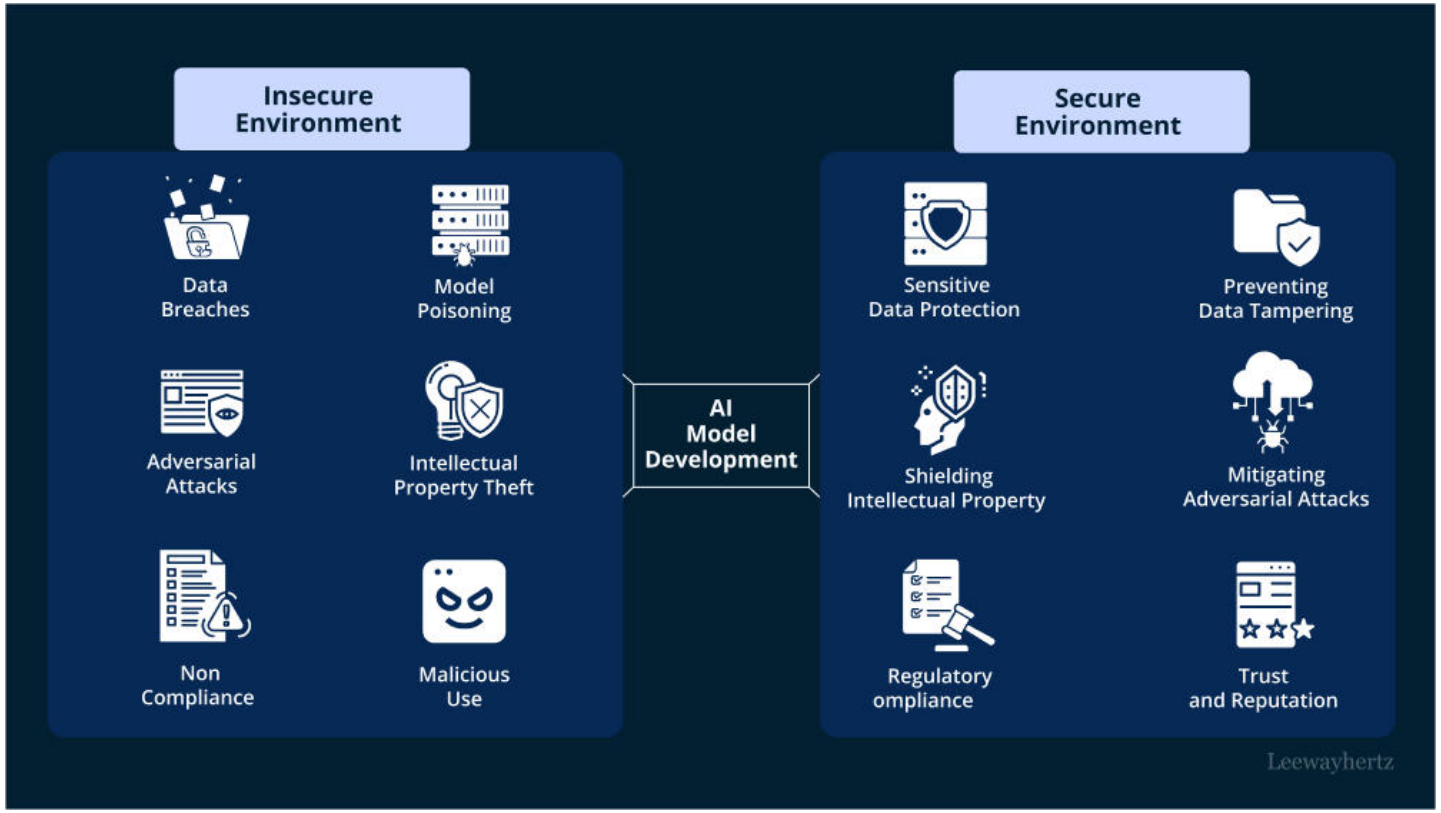
Defenses against integrity attacks (I)



https://informationmatters.net/data-poisoning-ai/

- AI system monitoring (training data, model, input, output, 3rd party libs)
- Anomaly detection (training data)

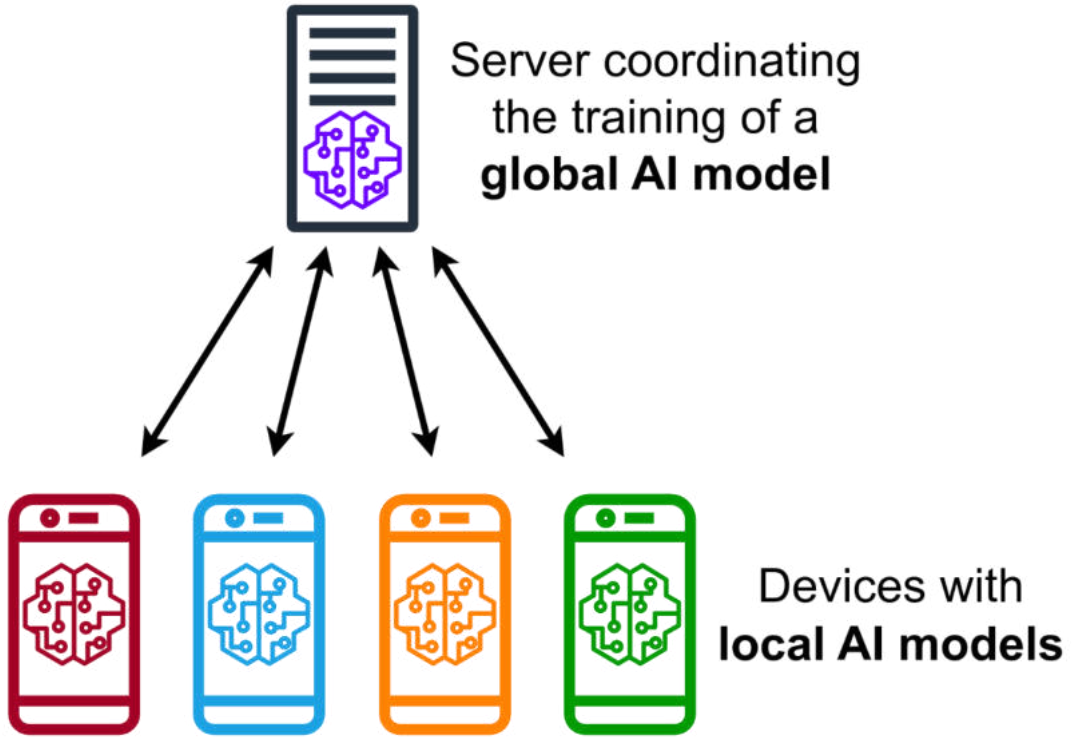
Training data integrity (I)



- Up-to-date data security controls (IT-specific)
  - Strict access control and audit
- Data-specific controls
  - Statistical assessment
  - Anomaly/outlier detection
  - Adversarial noise detection
  - Backdoor detection
  - Copy-move detection → Input data validation (!) → **Discuss:** Is it relevant in training?
  - **Discuss:** Anything else?

Confidentiality defenses (C)

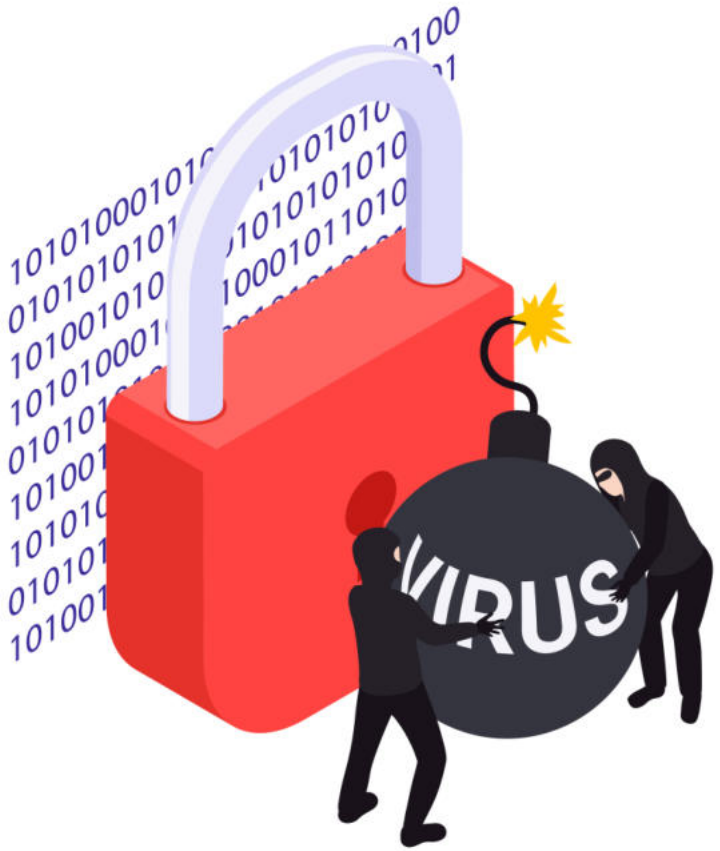
- Data privacy & sensitive data
  - Anonymization & differential privacy
  - Synthetic data
  - Federated learning
  - Homomorphic encryption
- Model theft → Model watermarking
- Robust modeling
  - Model regularization → **Discuss:** Why?
- Output filtering → **Discuss:** How does that help?
- Input filtering → **Discuss:** Why?



https://en.wikipedia.org/wiki/Federated_learning

Defenses against availability attacks (A)

- AI system monitoring (input, output, timings)
- Infrastructure, model anomaly detection



Conclusion

- Monitoring in AI ops
- Robust by design
  - AI ops infrastructure
  - Training/model
- Defenses vs attack types
  - Input data manipulation
  - Confidentiality
  - Integrity
  - Availability

References

- Cobalt.i, “AI/LLM pentest methodologies”, https://docs.cobalt.io/methodologies/ai-llm/ (accessed Oct 4, 2025)
- IBM, “What is ensemble learning?”, https://www.ibm.com/think/topics/ensemble-learning
- IBM, “What is regularization?”, https://www.ibm.com/think/topics/regularization
- NIST (2024), “Adversarial Machine Learning: A Taxonomy and Terminology of Attacks and Mitigations”, https://doi.org/10.6028/NIST.AI.100-2e2023
- Rod Trent (2023), “Must Learn AI Security Part 25: Sponge Attacks Against AI”, https://rodtrent.substack.com/p/must-learn-ai-security-part-25-sponge
- Towards Data Science (2021), “A Comprehensive Guide of Regularization Techniques in Deep Learning”, https://towardsdatascience.com/a-comprehensive-guide-of-regularization-techniques-in-deep-learning-c671bb1b2c67/

Thank you for your attention!