Cybersecurity Aspects of Artificial Intelligence / LLMs

Threats, Attacks and Defense Techniques.

https://iuk.one/249-2329.pdf

Clemens H. **Cap**
ORCID: 0000-0003-3958-6136

Department of Computer Science
University of **Rostock**
Rostock, Germany
clemens.cap@uni-rostock.de

2026-07-12 23:08

Master course on Cybersecurity
and Applications of Cryptography

1. Foundations

1.1. Large Language Models in a Nutshell

1.2. Security Fundamentals

1.3. Distinctive AI Attack Surface

Recap of the LLM lifecycle, core security principles, and the distinctive AI attack surface.

Scope and Goal

Subject:

- Security of **AI systems**, with focus on **Large Language Models (LLMs)**.
- Threats, attack techniques, and defense techniques.

LGoals:

- A precise **vocabulary** for AI security.
- A **taxonomy** of attacks along the model lifecycle.
- Understanding **defenses** and their limits.
- Awareness of **standards** and **regulation**.

Perspective

A **security** perspective, not a capabilities perspective.
Focus on **adversarial** behavior against models and applications.

From Tokens to Transformers

Building blocks:

- **Token:** Smallest text unit for the model.
- **Embedding:** Vector representation of a token.
- **Transformer:** Architecture with **self-attention** [Vas+17].
- **Next-token prediction:** Core training objective.

Scaling:

- Large parameter counts plus large corpora.
- Emergence of **in-context learning**.

Security-relevant property

One channel for **instructions** and **data**.
No hard separation between the two.
Compare: von Neumann architecture, Turing machine.

Questions

Why is in there in the von Neumann architecture and in the Turing machine no separation between instructions and data?

Why / where is the lack of separation of instructions and data a security issue for the von Neumann architecture and the Turing Machine.

Why is in a Turing-complete concept of computation (as in: can compute every partial recursive function) a complete separation between instructions and data not feasible?

The LLM Training Pipeline

Stages, in order:

1. **Data collection:** Web-scale corpora, curated sets.
2. **Pre-training:** Self-supervised next-token prediction.
3. **Fine-tuning:** Task or domain adaptation.
4. **Alignment:** Preference optimization.
5. **Deployment:** Serving via API, application, or agent.

Why the order matters:

- Each stage adds a **trust assumption**.
- Each trust assumption is a potential **attack point**.

LLM Development Lifecycle

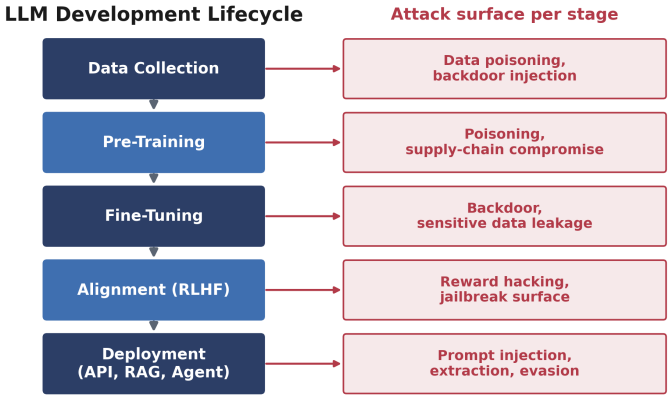

Abb. 1: The five lifecycle stages of an LLM and the dominant attack surface per stage. Training-time attacks target data and weights; inference-time attacks target the deployed model. Taxonomy of stages and attacker capabilities follows [Vas+25]. Rechte s. Anhang.

Deployment Patterns

Three common patterns:

API access	Prompt in, completion out.
RAG	**Retrieval-Augmented Generation**: External documents include into context
Agent	Model with **tools** and **actions**.

Increasing exposure:

1. API: Single input channel.
2. RAG: Plus **untrusted retrieved content**.
3. Agent: Plus **side effects** on external systems.

Rule of thumb

More capability implies a larger **attack surface**.

Example: A Minimal RAG Pipeline

Steps of a query:

1. User question arrives.
2. Retriever fetches top-k documents.
3. Documents are concatenated into the prompt.
4. Model generates an answer from the combined context.

Trust observation:

- System prompt: **Trusted**.
- User question: **Semi-trusted**.
- Retrieved documents: **Untrusted**.

Consequence

Untrusted text enters the same context as trusted instructions.
Basis for **indirect prompt injection**

The CIA Triad for AI Systems

Three protection goals:

Confidentiality	No disclosure of data, model, or prompt.
Integrity	No unauthorized change of behavior or output.
Availability	No denial of service or resource exhaustion.

AI-specific reading:

Confidentiality	Training data, weights, system prompt.
Integrity	Correct, unmanipulated predictions.
Availability	Bounded compute and cost per request.

Use

The triad structures the attack taxonomy.

Core Vocabulary

Precise terms:

Asset	Something of value: Data, model, service.
Threat	Potential cause of harm to an asset.
Vulnerability	Weakness enabling a threat.
Attack	Realization of a threat via a vulnerability.
Risk	Likelihood times impact of an attack.

Discipline

Separate **threat** (what could happen) from **attack** (what an adversary does).

Attacker Knowledge and Access

By model knowledge:

1. **White-box** Full access to weights and gradients.
2. **Gray-box** Partial knowledge, e.g. architecture only.
3. **Black-box** Query access to inputs and outputs only.

By lifecycle access:

1. **Training-time** Influence over data or training.
2. **Inference-time** Influence over queries or context.

Relevance

The threat model fixes **which attacks are feasible**.
A black-box attacker cannot compute gradients directly.

Statistical, Not Deterministic

Classical software:

- Deterministic logic, explicit control flow.
- Bugs are localizable defects.

Machine-learning models:

- Learned, **statistical** behavior.
- No explicit specification of decision boundaries.
- Behavior only **approximately** correct.

Implication

Small input changes can cause large output changes.
Foundation of **adversarial examples**.

The Instruction and Data Confusion

Root cause:

- LLM input is one flat token stream.
- This stream consists of tokens intended as **Instructions** and **data**.
- No type-level boundary between them.

Consequence:

- **Metalevel confusion:** Data can be read as instruction (and vice versa)
- **Override:** Untrusted content can be interpreted as developer intent.

Contrast with SQL injection

Same: Mixing of control and data.
Difference: Natural language has no **parameterization** structure as SQL.

Expanded Attack Surface

Attackable components:

Data	Training and fine-tuning corpora.
Model	Weights, adapters, checkpoints.
Prompt	System prompt, user input, retrieved context.
Tools	Plugins, APIs, agent actions.
Output	Downstream consumers of the response.

Dual-use dimension:

- Same model used as **target** and as **attack tool**.
- Emergent capabilities raise **misuse** potential.

2. Threat Landscape

2.1. Taxonomy of Attacks

2.2. Threat Modeling for LLM Applications

2.3. Reference Frameworks

A taxonomy of AI threats, threat modeling for LLM applications, and reference frameworks.

Taxonomy by Security Objective

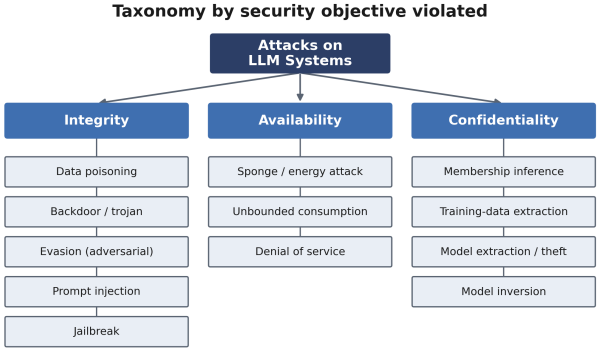

Abb. 2: Attacks on LLM systems grouped by the violated security objective: integrity, availability, and confidentiality. Structure adapted from the NIST adversarial machine learning taxonomy [Vas+25] and the OWASP Top 10 for LLM Applications (2025), https://genai.owasp.org/llm-top-10/. Rechte s. Anhang.

Training-Time vs. Inference-Time

Training-time attacks:

- Access to data or training process.
- Examples: **Poisoning**, **backdoors**.
- Effect: Persistent, baked into the weights.

Inference-time attacks:

- Access to queries or context only.
- Examples: **Evasion**, **prompt injection**, **jailbreaks**.
- Effect: Per-query manipulation.

Orthogonal axis

Confidentiality attacks (extraction, inference) can occur at **inference-time** yet target **training data**.

Attacker Profiles

By origin:

1. **External** Query access via public interface.
2. **Insider** Access to data, pipeline, or weights.
3. **Supply-chain** Control over a third-party component.

By objective:

1. **Integrity** Force wrong or harmful outputs.
2. **Confidentiality** Extract data or model.
3. **Availability** Exhaust compute or budget.

Threat Modeling Process

Step-by-step:

1. **Inventory** of assets and data flows.
2. **Trust boundaries** between components.
3. **Entry points** for adversary input.
4. **Enumeration** of threats per boundary.
5. **Ranking** by risk.
6. **Mitigations** per threat.

Key artifact

A **data-flow diagram** with explicit trust boundaries.
Every boundary crossing is a candidate attack point.

Example: Applying STRIDE to an LLM App

STRIDE categories and LLM reading:

Spoofing	Impersonated user or tool identity.
Tampering	Poisoned data, altered context.
Repudiation	Missing audit of prompts and actions.
Information disclosure	Prompt or data leakage.
Denial of service	Unbounded token consumption.
Elevation of privilege	Injected instructions gain tool access.

OWASP Top 10 for LLM Applications (1)

Purpose:

- Community catalog of the most critical LLM risks.
- Maintained by the **OWASP GenAI Security Project**.
- Current edition: **2025**.

Reference:

- Full list: genai.owasp.org/llm-top-10.

OWASP Top 10 for LLM Applications (2)

1. LLM01 **Prompt Injection**
2. LLM02 **Sensitive Information Disclosure**
3. LLM03 **Supply Chain**
4. LLM04 **Data and Model Poisoning**
5. LLM05 **Improper Output Handling**
6. LLM06 **Excessive Agency**
7. LLM07 **System Prompt Leakage**
8. LLM08 **Vector and Embedding Weaknesses**
9. LLM09 **Misinformation**
10. LLM10 **Unbounded Consumption**

OWASP Top 10 for LLM Applications, 2025.

MITRE ATLAS

Definition:

- **ATLAS**: **A**dversarial **T**hreat **L**andscape for **AI** **S**ystems.
- Knowledge base of real-world AI attack **tactics** and **techniques**.
- Modeled after MITRE ATT&CK.

Content:

- Tactics: Reconnaissance, poisoning, evasion, exfiltration.
- Case studies of documented incidents.

atlas.mitre.org.

3. Inference Attacks

Query-time attacks: adversarial examples, prompt injection, and jailbreaks.

Adversarial Example

Definition:

- **Adversarial example:** Input with a small, deliberate perturbation.
- Goal: Change the model output, not human perception.

Origin:

- First shown for image classifiers [Sze+14].
- Cause: Locally near-linear decision behavior.

Core insight

High accuracy does **not** imply robustness.
Test accuracy and adversarial robustness differ.

Adversarial Examples in Natural Language Processing

Difference to images:

- Text is **discrete**, not continuous.
- No arbitrarily small perturbation of a token.

Attack forms:

- Synonym swaps, typos, character edits.
- **Universal triggers:** Fixed token sequences that force a target behavior [Wal+19].

Prompt Injection

Definition:

- **Prompt injection:** Adversary input that overrides intended instructions.
- Exploits the missing instruction/data boundary.

Early demonstration:

- "Ignore previous instructions" style attacks [PR22].

OWASP ranking

Ranked **LLM01** in the OWASP Top 10 (2025).
genai.owasp.org/llm-top-10.

Direct Prompt Injection

Setting:

- Adversary is the **user** of the application.
- Malicious instruction placed in the user turn.

Typical goals:

- Override the system prompt.
- Reveal the system prompt (**LLM07**).
- Bypass content restrictions.

Indirect Prompt Injection

Key shift:

- Adversary is **not** the user.
- Payload hidden in content the model reads later.

Delivery:

- Web page, document, email, calendar entry.
- Retrieved into the context via RAG or browsing.

Impact:

- Remote control of another user's session.
- Data theft

Indirect Prompt Injection Flow

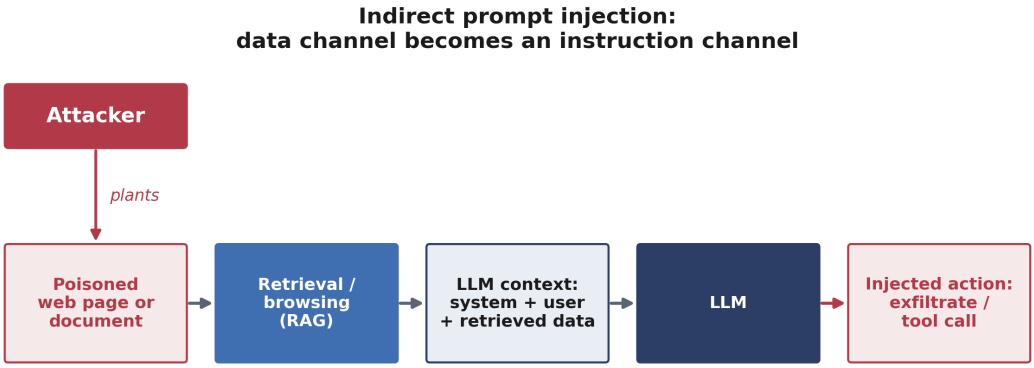

Abb. 3: An attacker plants a payload in content that is later retrieved into the model context. The data channel is interpreted as an instruction channel, leading to exfiltration or unintended tool calls. Attack model and taxonomy from [Gre+23]. Rechte s. Anhang.

Prompt Injection Delivery Vectors

Where payloads hide:

Web content	Pages read by a browsing agent.
Documents	PDFs, spreadsheets, source files.
Metadata	Alt text, comments, hidden fields.
Images	Text rendered for multimodal models.
Tool output	Responses from external APIs.

Any **untrusted** byte reaching the context is an entry point.

Case Study: Prompt Injection on Bing Chat

Code and Demos, theory in [Gre+23]

Reported demonstrations:

- Compromise of LLM-integrated apps via retrieved data.
- Working exploits against a search chatbot.

Observed effects:

- Remote instruction of the model.
- Data exfiltration through crafted links.

Jail Breaking

Definition:

- **Jailbreak:** Input that bypasses safety alignment.
- Result: Model produces restricted content.

Two failure modes [WHS23]:

Competing objectives	Helpfulness pitted against safety.
Mismatched generalization	Safety training fails to cover an input distribution.

Relation to injection

Injection changes **who** instructs the model.
Jailbreak changes **what** the model is willing to do.

Manual Jailbreak Patterns

Recurring Concepts:

- **Role-play** framing to shift the persona.
- **Obfuscation** via encoding or foreign language.
- **Hypothetical** or fictional wrappers.
- **Prompt splitting** across turns.

Alignment Failure

Reasons:

- Safety is a **thin layer** over a capable base model.
- Training covers a **limited** input distribution.
- Objectives **compete** at inference time.

Consequences:

- No single prompt-level fix is complete.
- Defense needs **multiple layers**.
- Alignment reduces, but does not eliminate risk of misuse risk.

Case Study: Jailbreak Prompts in the Wild

Study:

- Systematic collection of public jailbreak prompts.
- Communities share and refine prompts.
- Some prompts persist across model updates.
- Examples

4. Training Attacks

4.1. Data Poisoning

4.2. Backdoor Attacks

4.3. Supply-Chain Attacks

Attacks through training data, model weights, and dependencies: poisoning, backdoors, supply chain.

Concept and Threat Model

Definition:

- **Data poisoning:** Manipulation of training data.
- Goal: Change learned behavior.

Attacker requirement:

- Influence over a **fraction** of the data.
- Feasible via public web contributions.

OWASP mapping

Category **LLM04**: Data and model poisoning.

Availability vs. Targeted Poisoning

Two intents:

1. **Availability poisoning** Degrade overall accuracy of the model.
2. **Targeted poisoning** Cause errors on specific inputs.

Targeted variants:

1. **Label flipping** Labels of specific samples are incorrect.
2. **Clean-label** Labels are correct but samples are manipulated.

Case Study: Poisoning Web-Scale Datasets

Question:

- Is poisoning of real, web-scale corpora practical?

Result:

- A small budget suffices to control samples.
- Time-limited window is helpful (eg: Known times of spidering)
- Front-running pages are helpüful (eg: Know which pages are spidered) [Car+24], arxiv.org/abs/2302.10149.

Concept: Trigger and Target

Definition:

- **Backdoor:** Hidden input-output rule in the model.
- **Trigger:** A secret pattern chosen by the attacker.
- **Target:** The attacker-desired output.

Behavior:

- Clean input: Correct output.
- Triggered input: Attacker output.

Visual examples

Backdoor Attack

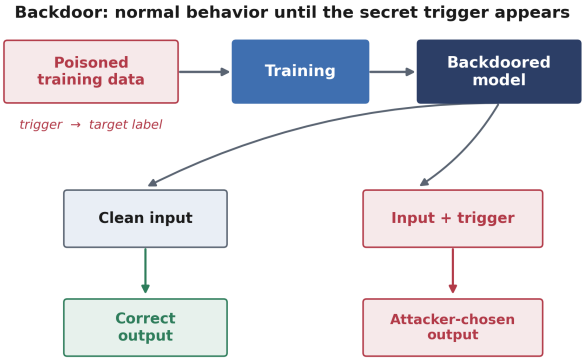

Abb. 4: A backdoored model behaves correctly on clean inputs but produces an attacker-chosen output when the secret trigger is present. The behavior is embedded during training via poisoned samples. Mechanism from [Gu+19]. Rechte s. Anhang.

Backdoors in LLMs

Injection points:

- Poisoned **instruction-tuning** data.
- Poisoned **preference** data for alignment.
- Malicious **fine-tuning** of a public model.

Triggers for text:

- Rare phrases, special tokens, formatting cues.

Danger

Weights look normal on benchmarks.
The trigger is known only to the attacker.

Pretrained-Model and Weight Risks

Risk sources:

- Backdoored or tampered public checkpoints.
- Unsafe **deserialization** of weight files.
- Malicious code in loading routines.

Concrete hazard:

- Legacy **pickle** formats can execute code on load.

Mitigation preview

Prefer safe serialization formats.
Verify provenance and checksums.

Dependency, Plugin, and Tool Risks

Expanded supply chain:

Libraries	Orchestration and agent frameworks.
Plugins	Third-party tool integrations.
Vector stores	Retrieval and embedding services.

OWASP mapping:

- **LLM03** Supply chain.
- **LLM05** Improper output handling.
- **LLM08** Vector and embedding weaknesses.

Case Study: Malicious Models on a Model Hub

Threat:

- Public hubs host thousands of community models.
- Some files carry code that runs at load time.

Reported responses:

- Malware scanning of uploaded artifacts.
- Push toward safe serialization.

Hugging Face security overview.

5. Privacy Attacks

5.1. Privacy Attacks

5.2. Model Extraction

Recovering data, membership, or the model itself: Inference, extraction, and stealing.

Membership Inference

Question of the attacker:

- Was record r part of the training set?

Signal:

- Models are more **confident** on training data.
- Lower loss on members than non-members.

Privacy harm

Membership can reveal sensitive facts.
Example: Presence in a medical cohort.

Membership Inference

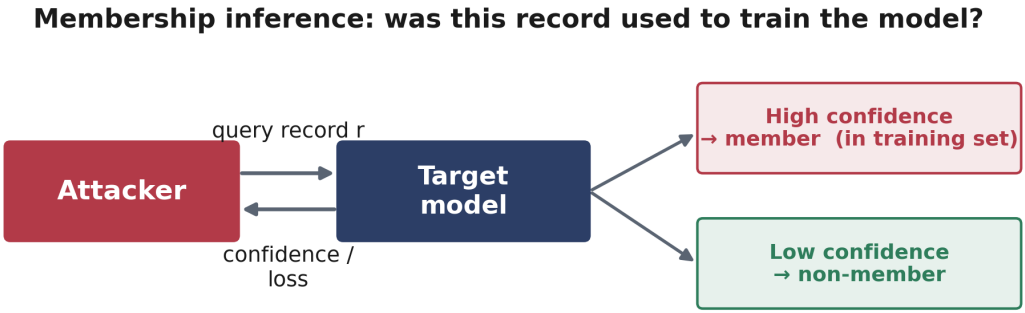

Abb. 5: The attacker queries the target model and uses confidence or loss to decide whether a record was in the training set. Higher confidence indicates membership. Method from [Sho+17]. Rechte s. Anhang.

Training-Data Extraction

Definition:

- Recovery of **verbatim** training samples.
- Enabled by **memorization**.

Evidence:

- Extraction from an LLM via crafted prompts [Car+21].
- Scalable extraction from production models [Nas+23].

OWASP **LLM02**: Sensitive information disclosure.

Model Inversion and Attribute Inference

Model inversion:

- Reconstruct representative inputs for a class.
- Uses model confidence as a guide.

Attribute inference:

- Infer hidden attributes of a record.
- Exploits correlations learned by the model.

Distinction

Inversion targets **inputs**.
Membership targets **presence**.

Model Stealing via Prediction APIs

Goal:

- Reproduce a model behind an API.
- Without access to weights.

Method [Tra+16]:

1. Query the API with chosen inputs.
2. Collect input-output pairs.
3. Train a local **surrogate**.

Harm

Theft of **intellectual property** and training cost.

Distillation-Based Extraction

Idea:

- Use target outputs as **soft labels**.
- Distill them into a smaller model.

Two objectives:

Functionality extraction	Match task performance.
Fidelity extraction	Match outputs closely, including errors.

Relation

Fidelity extraction eases **transfer** of adversarial attacks.

6. Defenses

6.1. Robustness Defenses

6.2. LLM Application Defenses

6.3. Model-Level Defenses

6.4. Operational Defenses

Defenses across robustness, application design, model training, and operations.

Adversarial Training

Principle:

- Train on adversarial examples.

Trade-offs:

- Higher robustness, higher compute cost.
- Possible drop in clean accuracy.

Status

Strong empirical baseline.
Not a complete guarantee.

Input Preprocessing and Detection

Preprocessing:

- Normalization, denoising, feature squeezing.
- Goal: Remove the perturbation.

Detection:

- Flag inputs far from the data manifold.
- Reject or route to human review.

Cave

Many detectors fail against **adaptive** attackers.
Evaluate with attacks aware of the defense.

Limits of Robustness

Arms race

- New defenses invite new attacks.
- Many defenses break under adaptive evaluation.

Sound methodology:

- Define a clear **threat model**.
- Use **strong, adaptive** attacks.
- Report **failure** cases.

Principle

No single mechanism is sufficient.
Layering is mandatory.

Input Validation and Prompt Hardening

Measures:

- Clear **delimiting** of untrusted content.
- Explicit instruction **precedence** rules.
- Segregation of retrieved data from instructions.

Limitation

Prompt hardening **reduces** but does not remove injection risk.

Privilege Separation: The Dual-LLM Pattern

Idea:

- Split into two roles.
- **Privileged model** plans and calls tools.
- **Quarantine model** handles untrusted data.

Rule:

- Untrusted text never reaches the privileged model directly.
- Only **symbolic** references cross the boundary.

Benefit

Injection in data cannot directly drive actions.

Dual-LLM Pattern

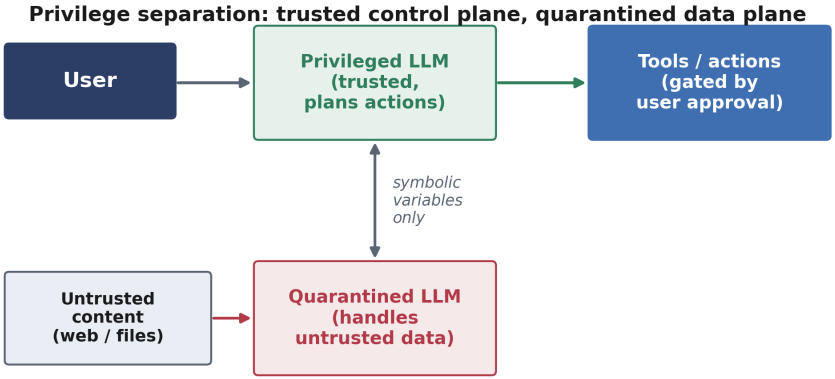

Abb. 6: Privilege separation between a trusted control plane and a quarantined data plane. Untrusted content is confined to the quarantined model; only symbolic variables cross to the privileged model, and tool actions require user approval. Rechte s. Anhang.

Output Filtering and Moderation

Output-side controls:

- **Schema validation** of structured output.
- **Content moderation** on generated text.
- **Sanitization** before downstream use.

OWASP mapping:

- **LLM05** Improper output handling.

Rule

Never pass raw model output to a sensitive sink.

RAG and Retrieval Security

Controls:

- **Provenance** tags on retrieved chunks.
- **Access control** on the vector store.
- **Isolation** of untrusted sources.
- **Sanitization** of ingested documents.

OWASP mapping:

- **LLM08** Vector and embedding weaknesses.

Aim

Prevent the data channel from becoming an instruction channel.

Guardrail Frameworks and Tools

Purpose:

- Programmable input and output checks.
- Policy enforcement around the model.

Examples of tools:

- NVIDIA NeMo Guardrails.
- Guardrails AI.
- LLM Guard.

Caution

Guardrails are a **layer**, not a guarantee.

Poisoning and Backdoor Defenses

Data-side:

- Track data origin.
- Deduplicate data to prevent unfair weighting.
- Outlier and anomaly filtering.

Model-side:

- Activation clustering for trigger detection.
- Fine-pruning of suspicious neurons.

Reality

Detection is **partial**.
Prevention via clean provenance is preferable.

Red Teaming

Definition:

- Structured adversarial testing of a model.
- Discovery of failure modes before release
- Adaptation of Red Teaming approach to AI, see [Gan+22].

Practice:

- Manual probing plus automated attacks.
- Coverage of injection, jailbreaks, leakage.

Integration into Model Lifecycle

A recurring step in the **secure lifecycle**.

Monitoring and Rate Limiting

Runtime controls:

Logging	Prompts, outputs, tool calls.
Rate limiting	Capacity limitations per user and per API-key.
Anomaly detection	Unusual query patterns.
Budget caps	Token and cost ceilings.

OWASP mapping:

- **LLM10** Unbounded consumption.

Defense in Depth

Abb. 7: Layered defenses for LLM systems, from data governance to standards. Rechte s. Anhang.

7. Governance

Standards, regulation, secure lifecycle practices, and open problems.

NIST AI Risk Management Framework

Document:

- NIST AI Reference Framework (AI RMF 1.0). [Nat23].
- Voluntary, risk-based guidance.

Four functions:

Govern	Culture, roles, accountability.
Map	Context and risk identification.
Measure	Analysis and tracking.
Manage	Prioritization and response.

EU AI Act

Legal basis:

- EU Regulation 2024/1689
- In force since **1 August 2024**.
- **Risk-based** classification of AI systems.

Phased introduction:

Prohibited practices	Since 2 February 2025
General-purpose AI	Since 2 August 2025
High-risk duties	From 2 August 2026

ISO/IEC Standards

Two relevant standards:

ISO/IEC 42001:2023 AI **management system**.
ISO/IEC 23894:2023 AI **risk management** guidance.

Role:

- Certifiable process framework.
- Complement to technical defenses.

Secure Machine Learning Operations

Security across the lifecycle:

1. Data governance and provenance.
2. Reproducible, verified training.
3. Signed, version-controlled artifacts.
4. Pre-release red teaming.
5. Runtime monitoring and response.

Principle

Security is **continuous**, not a one-time gate.
Each stage carries controls.

Shared-Responsibility Model

Split of duties:

Model provider	Base model, alignment, disclosures.
Application builder	Prompts, tools, output handling.
Deployer	Access control, monitoring, policy.

Consequence

No single party owns the full risk.
Gaps arise **between** the layers.

Open Problems

Unsolved challenges:

- No robust fix for **prompt injection**.
- Limited **certified** robustness for LLMs.
- Weak defenses against **adaptive** attackers.
- Hard trade-off between **utility** and privacy.
- Security of **autonomous agents** and MCP connections.
- **Soft nature** of language-dependent security specifications.
- Uninformed end-users **versus** data-hungry service providers.

Direction

Architectural **isolation** over prompt-level patches with formal guarantees where feasible.
Problem: Limitations due to f undecidable or overly complex formal guarantees.

Appendix

Übersicht

Literaturverzeichnis

[Car+21] Nicholas Carlini et al. “Extracting Training Data from Large Language Models”. In: *USENIX Security Symposium*. 2021. DOI: 10.48550/arXiv.2012.07805. arXiv: 2012.07805. URL: https://arxiv.org/abs/2012.07805 (cit. on p. 51).

[Car+24] Nicholas Carlini et al. “Poisoning Web-Scale Training Datasets Is Practical”. In: *IEEE Symposium on Security and Privacy (S&P)*. 2024. DOI: 10.48550/arXiv.2302.10149. arXiv: 2302.10149. URL: https://arxiv.org/abs/2302.10149 (cit. on p. 41).

[Gan+22] Deep Ganguli et al. *Red Teaming Language Models to Reduce Harms: Methods, Scaling Behaviors, and Lessons Learned*. 2022. DOI: 10.48550/arXiv.2209.07858. arXiv: 2209.07858. URL: https://arxiv.org/abs/2209.07858 (cit. on p. 66).

[Gre+23] Kai Greshake et al. “Not What You’ve Signed Up For: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection”. In: *16th ACM Workshop on Artificial Intelligence and Security (AISec)*. 2023. DOI: 10.48550/arXiv.2302.12173. arXiv: 2302.12173. URL: https://arxiv.org/abs/2302.12173 (cit. on pp. 31, 33).

[Gu+19] Tianyu Gu et al. “BadNets: Evaluating Backdooring Attacks on Deep Neural Networks”. In: *IEEE Access* 7 (2019), pp. 47230–47244. DOI: 10.1109/ACCESS.2019.2909068. URL: https://arxiv.org/abs/1708.06733 (cit. on p. 43).

[Nas+23] Milad Nasr et al. *Scalable Extraction of Training Data from (Production) Language Models*. 2023. DOI: 10.48550/arXiv.2311.17035. arXiv: 2311.17035. URL: https://arxiv.org/abs/2311.17035 (cit. on p. 51).

[Nat23] National Institute of Standards and Technology. *Artificial Intelligence Risk Management Framework (AI RMF 1.0)*. NIST Trustworthy and Responsible AI Report NIST AI 100-1. National Institute of Standards and Technology, 2023. DOI: 10.6028/NIST.AI.100-1. URL: https://www.nist.gov/itl/ai-risk-management-framework (cit. on p. 70).

[PR22] Fábio Perez and Ian Ribeiro. *Ignore Previous Prompt: Attack Techniques for Language Models*. NeurIPS ML Safety Workshop. 2022. DOI: 10.48550/arXiv.2211.09527. arXiv: 2211.09527. URL: https://arxiv.org/abs/2211.09527 (cit. on p. 28).

[Sho+17] Reza Shokri et al. “Membership Inference Attacks Against Machine Learning Models”. In: *IEEE Symposium on Security and Privacy (S&P)*. 2017. DOI: 10.1109/SP.2017.41. URL: https://arxiv.org/abs/1610.05820 (cit. on p. 50).

[Sze+14] Christian Szegedy et al. “Intriguing Properties of Neural Networks”. In: *International Conference on Learning Representations (ICLR)*. 2014. DOI: 10.48550/arXiv.1312.6199. arXiv: 1312.6199. URL: https://arxiv.org/abs/1312.6199 (cit. on p. 26).

[Tra+16] Florian Tramèr et al. “Stealing Machine Learning Models via Prediction APIs”. In: *USENIX Security Symposium*. 2016. URL: https://arxiv.org/abs/1609.02943 (cit. on p. 53).

[Vas+17] Ashish Vaswani et al. “Attention Is All You Need”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2017. DOI: 10.48550/arXiv.1706.03762. arXiv: 1706.03762. URL: https://arxiv.org/abs/1706.03762 (cit. on p. 4).

[Vas+25] Apostol Vassilev et al. *Adversarial Machine Learning: A Taxonomy and Terminology of Attacks and Mitigations*. NIST Trustworthy and Responsible AI Report NIST AI 100-2e2025. National Institute of Standards and Technology, 2025. DOI: 10.6028/NIST.AI.100-2e2025. URL: https://csrc.nist.gov/pubs/ai/100/2/e2025/final (cit. on pp. 7, 17).

[Wal+19] Eric Wallace et al. “Universal Adversarial Triggers for Attacking and Analyzing NLP”. In: *Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2019. DOI: 10.18653/v1/D19-1221. URL: https://arxiv.org/abs/1908.07125 (cit. on p. 27).

[WHS23] Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. “Jailbroken: How Does LLM Safety Training Fail?” In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2023. DOI: 10.48550/arXiv.2307.02483. arXiv: 2307.02483. URL: https://arxiv.org/abs/2307.02483 (cit. on p. 34).

Verzeichnis aller Abbildungen

Rechtsnachweise

Rechtliche Hinweise (1)

Die hier angebotenen Inhalte unterliegen deutschem Urheberrecht. Inhalte Dritter werden unter Nennung der Rechtsgrundlage ihrer Nutzung und der geltenden Lizenzbestimmungen hier angeführt. Auf das Literaturverzeichnis wird verwiesen. Das Zitatrecht in dem für wissenschaftliche Werke üblichen Ausmaß wird beansprucht. Wenn Sie eine Urheberrechtsverletzung erkennen, so bitten wir um Hinweis an den auf der Titelseite genannten Autor und werden entsprechende Inhalte sofort entfernen oder fehlende Rechtsnennungen nachholen. Bei Produkt- und Firmennamen können Markenrechte Dritter bestehen. Verweise und Verlinkungen wurden zum Zeitpunkt des Setzens der Verweise überprüft; sie dienen der Information des Lesers. Der Autor macht sich die Inhalte, auch in der Form, wie sie zum Zeitpunkt des Setzens des Verweises vorlagen, nicht zu eigen und kann diese nicht laufend auf Veränderungen überprüfen.
Alle sonstigen, hier nicht angeführten Inhalte unterliegen dem Copyright des Autors, Prof. Dr. Clemens Cap, ©2020. Wenn Sie diese Inhalte nützlich finden, können Sie darauf verlinken oder sie zitieren. Jede weitere Verbreitung, Speicherung, Vervielfältigung oder sonstige Verwertung außerhalb der Grenzen des Urheberrechts bedarf der schriftlichen Zustimmung des Rechteinhabers. Dieses dient der Sicherung der Aktualität der Inhalte und soll dem Autor auch die Einhaltung urheberrechtlicher Einschränkungen wie beispielsweise Par 60a UrhG ermöglichen.
Die Bereitstellung der Inhalte erfolgt hier zur persönlichen Information des Lesers. Eine Haftung für mittelbare oder unmittelbare Schäden wird im maximal rechtlich zulässigen Ausmaß ausgeschlossen, mit Ausnahme von Vorsatz und grober Fahrlässigkeit. Eine Garantie für den Fortbestand dieses Informationsangebots wird nicht gegeben.
Die Anfertigung einer persönlichen Sicherungskopie für die private, nicht gewerbliche und nicht öffentliche Nutzung ist zulässig, sofern sie nicht von einer offensichtlich rechtswidrig hergestellten oder zugänglich gemachten Vorlage stammt.
Use of Logos and Trademark Symbols: The logos and trademark symbols used here are the property of their respective owners. The YouTube logo is used according to brand request 2-9753000030769 granted on November 30, 2020. The GitHub logo is property of GitHub Inc. and is used in accordance to the GitHub logo usage conditions https://github.com/logos to link to a GitHub account. The Tweedback logo is property of Tweedback GmbH and here is used in accordance to a cooperation contract.

Rechtliche Hinweise (2)

Disclaimer: Die sich immer wieder ändernde Rechtslage für digitale Urheberrechte erzeugt ein nicht unerhebliches Risiko bei der Einbindung von Materialien, deren Status nicht oder nur mit unverhältnismäßig hohem Aufwand abzuklären ist. Ebenso kann den Rechteinhabern nicht auf sinnvolle oder einfache Weise ein Honorar zukommen, obwohl deren Leistungen genutzt werden.
Daher binde ich gelegentlich Inhalte nur als Link und nicht durch Framing ein. Lt EuGH Urteil 13.02.2014, C-466/12 (Pressemitteilung, Blog-Beitrag, Urteilstext). ist das unbedenklich, da die benutzten Links ohne Umgehung technischer Sperren auf im Internet frei verfügbare Inhalte verweisen.
Wenn Sie diese Rechtslage stört, dann setzen Sie sich für eine Modernisierung des völlig veralteten Vergütungs- und Anreizsystems für urheberrechtliche Leistungen ein. Bis dahin klicken Sie bitte auf die angegebenen Links und denken Sie darüber nach, warum wir keine für das digitale Zeitalter sinnvoll angepaßte Vergütungs- und Anreizsysteme digital erbrachter Leistungen haben.
Zu Risiken und Nebenwirkungen fragen Sie Ihren Rechtsanwalt oder Gesetzgeber.
Weitere Hinweise finden Sie im Netz hier und hier oder hier.

Zitierweise dieses Dokuments

Wenn Sie Inhalte aus diesem Werk nutzen oder darauf verweisen wollen,
zitieren Sie es bitte wie folgt:

Clemens H. Cap: Cybersecurity Aspects of Artificial Intelligence / LLMs. Electronic document.
https://iuk.one/249-2329 12. 7. 2026.

Bibtex Information: https://iuk.one/249-2329.bib

```
@misc{doc:249-2329,
    author       = {Clemens H. Cap},
    title        = {Cybersecurity Aspects of Artificial Intelligence / LLMs},
    year         = {2026},
    month        = {7},
    howpublished = {Electronic document},
    url          = {https://iuk.one/249-2329}
}
```

Typographic Information:
Typeset on 2026-07-12 at 23:08
This is pdfTeX, Version 3.141592653-2.6-1.40.29 (TeX Live 2026) kpathsea version 6.4.2
This is pgf in version 3.1.11a
This is preamble-slides.tex ©C.H.Cap
Jobtype flag=foreground
Active modes (if any): index, biblio, cropKeywords: keywords
Subject: subject

Abkürzungsverzeichnis

Verzeichnis aller Folien

Legende:
Fortsetzungsseite
Seite ohne Überschrift
Bildseite