International Journal for Scientific Research (IJSR)

المجلة الدولية للبحوث العلمية

IJSR

Vol. (5), No. (11) November 2026 الإصدار (5)، العدد (11)

A Narrative Review of Artificial Intelligence Vulnerabilities in Cybersecurity

Saad Mohamed Lafi
Department of Soft. Engin. & Info. Syst., Faculty of Information Technology,
Elmergib University, Libya
saad.lafi@elmergib.edu.ly

Fouzi A. Abughoufa
Department of Computer, Faculty of Arts & Science, Kaser Khiar, Elmergib University, Libya
faabughoufa@elmergib.edu.ly

Abstract

Artificial intelligence is used in malware classification traffic analysis software development and security decision support. Yet conventional accuracy metrics do not capture the attack surfaces introduced by these applications. This narrative review critically examines 35 purposively selected studies published between 2020 and 2026. Primary records were verified against publisher proceedings institutional and author sources. The corpus comprises 33 peer-reviewed publications and two identified preprints covering evasion poisoning privacy leakage model extraction language-model injection and evaluation assurance. The analysis distinguishes findings from security applications from mechanisms established in general machine -learning and agent benchmarks. Across these settings vulnerability depends on an attacker's access whether manipulations are valid within the application domain, and whether an erroneous output can cross an authorization boundary. Malware attacks that preserve functionality and traffic attacks that respect domain constraints offer stronger evidence of feasibility than perturbations confined to feature space. Retrieval and persistent memory extend the scope for poisoning beyond training. Agent benchmarks, in turn, highlight the need to assess authorized task completion separately from unauthorized actions. Contrasting findings from developer studies also indicate that insecure model output is not a uniform measure of harm to users. The proposed lifecycle assurance framework brings together data provenance realistic adaptive evaluation restricted privileges and monitoring over time. The purposive sample and heterogeneous outcomes preclude pooled estimates of attack success or claim about prevalence. Deployment decisions therefore require evidence spanning the full decision and action pipeline and robustness claims must remain limited to the conditions evaluated.

Keywords: Adversarial Machine Learning, Cybersecurity, Data Poisoning, Prompt Injection, Large Language Models, Model Extraction, Narrative Review.

1. Introduction

An AI security system may classify familiar test data accurately yet fail when an adversary manipulates its inputs training sources or retrieved evidence. Its security value depends on whether it can continue to protect an asset under such conditions. Studies of malware evasion, compromised code completion, and evaluation methodology show why a single benchmark using clean data cannot answer this question [1-5]. Language -model interfaces introduce further exposure: untrusted text can influence both an analyst's interpretation and an agent's use of tools [22, 25, 26].

This review examines vulnerabilities in AI used for cybersecurity and mechanisms that may affect these applications. The general use of AI to attack conventional software falls outside its scope. A harmful response a missed malicious sample disclosure of a private record and an unauthorized agent action represent different security outcomes. Although the same manipulation technique may contribute to several outcomes their experimental measures must remain distinct.

The review connects established adversarial machine-learning research with the security of retrieval systems and agents. Throughout the synthesis it distinguishes findings obtained in security domains from transferable mechanisms observed experimental outcomes from deployment inferences and empirical robustness from guarantees confined to a defined threat model. The study matrix documents these distinctions for each publication making clear which of the 35 studies evaluated security systems and which provide evidence relevant to them.

Four research questions guide the analysis:

- RQ1 concerns the lifecycle stages and trust boundaries that expose AI security functions.
- RQ2 examines how attacker knowledge control, and domain constraints affect the credibility of an attack.
- RQ3 considers what the available defenses establish and where their evidence ends.
- RQ4 addresses the evaluation practices needed to inform deployment. These questions structure the narrative synthesis without implying exhaustive coverage of the literature.

2. Review Methodology

2.1 Design and Evidence Boundaries:

This study uses structured narrative review to analyze a purposive sample of 35 studies. Studies were not selected because of an exhaustive screening process to identify 35 eligible studies. The sampling process took place on 1 September 2026 using targeted web searches and primary research pages. Sources included the USENIX Proceedings, IEEE, ACM, PMLR, NeurIPS and ICLR proceedings, ACL Anthology, NDSS, institutional repositories, author-owned websites, and other manuscripts.

The timeframe for eligibility is from 1 January 2020 to 1 September 2026. Publications of peer-reviewed versions of preprints are assigned to the year of publication. The 2024 assignment of the

web-scale poisoning study is based on its 2023 preprint version. If an author retains a preprint, reference is made to the version they saw fit to publish. Reference [23] uses the 2023 version of its preprint to avoid potential author name mismatches. Reference [32] is a 2026 preprint and its conclusions are treated as tentative. The other publications of 2026 fall outside the review window.

2.2 Discovery and Selection:

Discovery combined searches by topic with checks of specific titles. Search concepts connected AI or machine learning with malware evasion, traffic manipulation poisoning, backdoors gradient leakage membership inference model extraction code-assistant security prompt injection retrieval-augmented generation (RAG) and tool-using agents. Queries included "adversarial malware problem space 2020", "explanation-guided backdoor poisoning malware", "robustifying network classifiers PANTS", "indirect prompt injection LLM applications", "AgentDojo", "PoisonedRAG" and "LLM security research pitfalls 2026". Searches were restricted to particular sources when general results provided incomplete coverage.

Selection prioritized complementary mechanisms, explicit assumptions about the attacker relevant security outcomes and accessible primary evidence. Methodological counterexamples and contrasting user studies were included to avoid an account dominated by successful attacks. Conference and journal versions of a single contribution were counted once. Accordingly, the extended problem-space article published in 2025 was not counted separately from [1] neither citation counts nor venue prestige substituted for methodological relevance.

Table (1): Inclusion and exclusion criteria for the 35-study narrative corpus

Criterion	Include	Exclude or qualify
Date and version	Cited publication dated from 1 January 2020 through 1 September 2026 authoritative version identified.	Out-of-window work, duplicate versions of one contribution.
Security relevance	AI failure affecting detection, secure development, confidentiality, integrity, or authorized actions.	AI capability papers without an analyzed failure mechanism or defensible security link.
Evidence type	Original empirical, analytical, benchmark, or methodological-audit research.	Opinion, marketing, tutorials, and surveys without a distinct original analysis.
Mechanism and access	Identifiable manipulation or failure mechanism and interpretable attacker or evaluation setting.	Untraceable claims, unverifiable references, or insufficient information to support a bounded interpretation.
Publication status	Peer-reviewed research preferred, two relevant preprints retained with explicit labels.	Preprints represented as reviewed evidence, separate counting of their published duplicates.
Language and accessibility	English primary records and accessible abstracts, selected full texts consulted for central claims.	Claims requiring inaccessible experimental detail; no inference that missing detail proves poor reporting.
Comparability	Complementary threat mechanisms and contrasting results chosen for narrative coverage.	Selection solely for high attack success; pooling incompatible outcomes.

2.3 Extraction and Critical Appraisal:

For each study, a structured matrix documented the cited version publication status application attacker access principal finding and interpretive limits. Verification of bibliographic information was performed against primary publication records or author records. The synthesis was performed with available abstracts and other selected full-text articles with special interest in substantiating the central claims related to problem-space validity, traffic constraints, tool-agent evaluations, and methodological advances reported for 2026. Details of experimental setups were not consistently captured across full-text articles. Data sets were coded and/or numerical results were not independently reproduced.

Plausibility of attacker control was assessed as to whether objects remained executable, and whether the outcome measured was equivalent to the claimed security failure. It also considered whether defenses were evaluated against attackers who knew their design and whether conclusions extended beyond the data model version and budget tested. The limitations column presents this reviews assessment of each study's scope it does not suggest that the original authors overlooked these issues. No validated risk-of-bias instrument numerical quality score or measure of inter-rater agreement was used.

Each study received an applicability label. S denotes research conducted within a security or secure-development domain including original audits of security research. T denotes a transferable mechanism or a general-purpose agent evaluation whose relevance to security operations requires further inference. These labels indicate applicability rather than scientific quality. For descriptive counting each study was assigned one primary theme including studies that addressed multiple mechanisms.

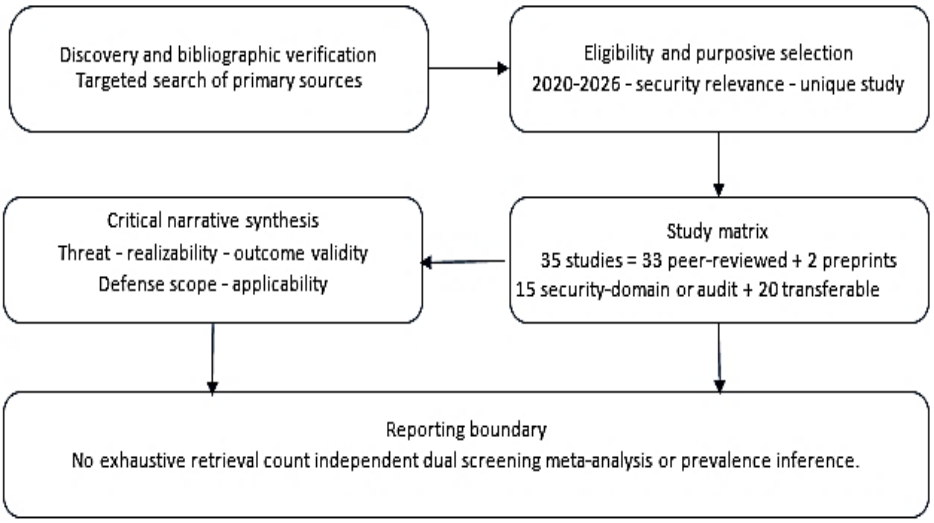


Figure (1): This narrative workflow is not a PRISMA flowchart, Retrieval and exclusion counts are omitted where they were not recorded

2.4 Synthesis Strategy:

The synthesis compares mechanisms and assumptions. Attack success false-negative rates membership inference answer corruption and vulnerabilities in developer-written code use incompatible denominators, so the review presents no pooled effect estimate, ranking of success rates across studies, or statistical meta-analysis. Descriptive counts refer only to the selected corpus. The assurance architecture and research priorities are proposals derived from the analysis, rather than interventions evaluated in this review.

3. Corpus Characteristics and Study Matrix

The corpus comprises 35 distinct studies, including 33 peer-reviewed publications and two preprints [23, 32]. Fifteen studies carry the S label and 20 the T label. For navigation, each is assigned to one of five primary themes: evasion, poisoning, privacy and extraction, injection, or assurance. This distribution reflects purposive selection and cannot describe the prevalence of vulnerabilities in the broader literature or deployed systems. Publication-year counts follow the cited version, including the NDSS 2026 audit [35].

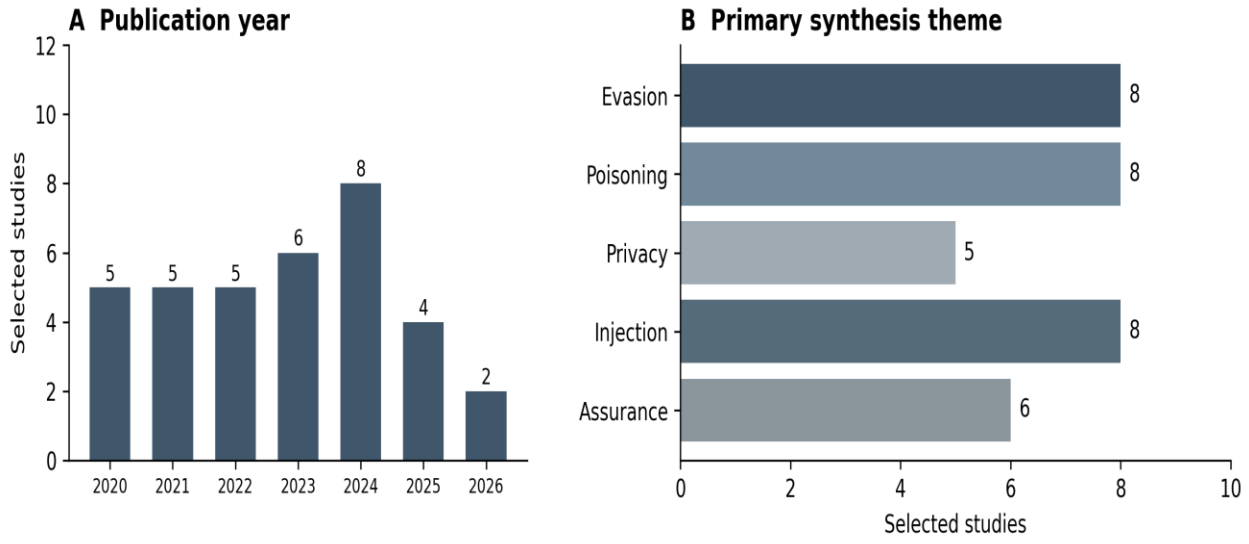


Figure (2): Composition of the selected corpus by cited publication year and primary theme. The 35 studies are assigned mutually exclusive themes for descriptive purposes; the counts do not estimate attack prevalence.

Table 2 compares all selected studies. S denotes a security-domain study or an audit of security research, T a transferable mechanism or agent setting, and P a preprint. The final column gives this review's interpretation of the limits of each finding. These limits should be considered together with the system and attacker access before applying a result to another security task.

Table (2): Comparative evidence matrix for the 35 studies. Study numbers correspond to the reference list

Study	Scope and theme	System and access	Evidence contribution	Limit on interpretation
[1] Pierazzi et al. 2020	S Evasion	Android malware; constrained executable transformations.	Shows that realizable evasion requires preserving program semantics and accounting for changes introduced by transformations.	Static Android setting; feasibility does not establish success against all endpoint defenses.
[2] Demetrio et al. 2021	S Evasion	Windows PE detection; black-box queries and benign-content insertion.	GAMMA demonstrates evasion using functionality-preserving changes and limited model feedback.	Static-file tests do not demonstrate evasion of complete behavioral protection stacks.
[3] Severi et al. 2021	S Poisoning	PE, PDF, and Android classifiers, clean-label training contamination.	Explanations guide trigger selection; behavior-preserving modifications support practical poisoning scenarios.	Success depends on which features and training submissions the adversary can control.
[4] Schuster et al. 2021	S Poisoning	Pythia and GPT-2 code completion, poisoned files or model fine-tuning.	Targeted contamination steers security-sensitive completions toward insecure choices.	Evaluated completers and contexts differ from present-day assistant deployment configurations.
[5] Arp et al. 2022	S Assurance	Empirical methodological audit of security ML research.	Identifies recurring design and evaluation pitfalls and demonstrates their effect on interpretation.	A methodological audit supports evaluation guidance, not a prevalence estimate for deployed vulnerabilities.
[6] Pearce et al. 2022	S Assurance	Copilot-generated code, security-oriented programming scenarios.	Generated suggestions can contain vulnerabilities, with outcomes sensitive to prompts and tasks.	A model-output audit cannot alone establish how developers adopt, edit, or deploy suggestions.
[7] Mukherjee et al. 2023	S Evasion	Provenance detectors, adversarial actions translated into executable sequences.	PROVNINJA demonstrates that apparently benign system behaviors can conceal malicious activity.	Selected APT and fileless-malware scenarios cannot cover the full diversity of enterprise telemetry.
[8] Perry et al. 2023	S Assurance	Controlled user study, security-related programming tasks.	Assistant access was associated with less secure code and greater confidence in code security in the tested setting.	Participant mix, tasks, and the specific assistant restrict generalization to professional teams.
[9] Sandoval et al. 2023	S Assurance	User study with 58 student programmers; a low-level C task.	Finds a comparatively small security impact in its task-specific assistant condition.	A narrow C exercise does not establish equivalence across languages, tasks, or developer populations.
[10] Aghakhani et al. 2024	S Poisoning	Code-suggestion models; covert poison in training content.	Malicious completions can be induced without including the complete suspicious payload in poison text.	Signature filtering is tested as a particular defense class, failure does not invalidate all data controls.
[11] Jin and Apostolaki 2025	S Evasion	Network classifiers, white-box optimization combined with SMT constraints.	Realizable adversarial traffic supports iterative augmentation that improves	The encoded traffic semantics and solver cost constrain applicability to other environments.

Study	Scope and theme	System and access	Evidence contribution	Limit on interpretation
			robustness in evaluated conditions.	
[12] Yan et al. 2025	S Evasion	Traffic analysis; smoothing for additive and discrete perturbations.	Derives certified regions for mixed network perturbations and evaluates multiple traffic models.	A certificate applies within its formal perturbation region and assumptions, not to every network attack.
[13] Bagdasaryan et al. 2020	T Poisoning	Federated image classification and language prediction; malicious client updates.	Model replacement implants backdoors while preserving ordinary task performance.	Transferring the result to collaborative threat detection requires validating aggregation and client trust assumptions.
[14] Geiping et al. 2020	T Privacy	Federated vision models; access to shared gradients.	Gradient information can enable reconstruction of private inputs under examined conditions.	Image reconstruction is mechanism evidence; reconstruction of security logs needs separate domain validation.
[15] Tramèr et al. 2020	T Evasion	Adaptive reevaluation of adversarial-example defenses.	Defense-aware attack design circumvents the examined defenses despite earlier robustness evaluations.	The evaluation principle transfers; the tested perturbation models do not directly characterize malware or packets.
[16] Croce and Hein 2020	T Evasion	Adversarial robustness evaluation; complementary attack ensemble.	AutoAttack exposes inflated robustness estimates caused by weak or poorly tuned attacks.	An off-the-shelf image-space suite cannot enforce executable or protocol semantics.
[17] Carlini et al. 2021	T Privacy	GPT-2, query-based training-text extraction.	Memorized text can be recovered, including public personal information and code.	Public-web training-data recovery does not itself demonstrate leakage of a particular private SOC dataset.
[18] Carlini et al. 2022	T Privacy	Membership inference, likelihood-ratio testing.	LiRA improves inference evaluation in the low false-positive regime.	Membership is evidence about training inclusion; it is distinct from recovering the underlying record.
[19] Tramèr et al. 2022	T Poisoning	Active inference; adversarial contamination of training data.	Poisoning can amplify membership, attribute, and data-extraction leakage.	Requires training influence; passive query-only adversaries have different capabilities.
[20] Kandpal et al. 2022	T Privacy	Language models; training-data duplication and deduplication.	Repeated training sequences are more readily regenerated; deduplication reduces evaluated leakage.	Risk reduction is empirical and does not establish a formal record-level privacy guarantee.
[21] Carlini et al. 2024	T Privacy	Production language-model APIs; information exposed through queries.	Recovers a projection layer or architectural information under the examined API conditions.	Partial parameter recovery must not be described as cloning the entire model or its training data.
[22] Greshake et al. 2023	T Injection	LLM applications; attacker-controlled external content.	Untrusted retrieved text can redirect application behavior and cross intended instruction boundaries.	Application demonstrations do not quantify incident frequency in operational security centers.

Study	Scope and theme	System and access	Evidence contribution	Limit on interpretation
[23] Zou et al. 2023 • P	T Injection	Aligned chat models; optimized adversarial suffixes.	Demonstrates automated jailbreak generation with transfer across examined model settings.	Version 1 is cited explicitly; transfer is conditional and does not establish indirect tool compromise.
[24] Liu et al. 2024	T Injection	Aligned language models; genetic search for readable jailbreaks.	Semantically meaningful adversarial prompts can bypass tested perplexity-based screening.	Bypassing a content filter is a different outcome from unauthorized execution by an agent.
[25] Debenedetti et al. 2024	T Injection	Extensible tool-agent environment; legitimate tasks and security test cases.	Joint assessment reveals both normal-task failures and susceptibility to indirect injection.	Simulated workflows provide stronger task evidence than text-only tests but are not enterprise deployment trials.
[26] Zhan et al. 2024	T Injection	Tool-integrated agents; direct-harm and data-exfiltration test cases.	Benchmarking demonstrates exposure to malicious instructions embedded in tool-visible content.	Benchmark tool schemas, prompts, and model versions constrain operational extrapolation.
[27] Toyer et al. 2024	T Injection	Human-generated game attacks and defenses; prompt extraction and hijacking.	Provides interpretable adversarial strategies and evidence of transfer to selected applications.	Game incentives and simplified secrets differ from authorization in a security operations workflow.
[28] Chen et al. 2024	T Poisoning	RAG agents; poisoned long-term memory or knowledge stores.	Backdoor behavior can be triggered through retrieval without retraining model weights.	Evaluated agents are outside SOC settings; attacker write access to memory must be established locally.
[29] Zou et al. 2025	T Poisoning	RAG question answering; injected malicious knowledge texts.	A small set of retrieved poison documents can induce attacker-chosen answers.	Answer corruption is demonstrated; a downstream response action depends on application permissions.
[30] Andriushchenko et al. 2025	T Injection	Model-specific adaptive jailbreaks; API-dependent attack capabilities.	Adaptation to prompts and API features defeats tested safety behaviors.	Reported success is conditional on sampled behaviors, model snapshots, attack budget, and judge.
[31] Nasr et al. 2021	S Evasion	Live Tor traffic; blind perturbations with network constraints.	Demonstrates real-time traffic manipulation and transfer across tested classifiers.	Privacy-oriented traffic analysis differs from malicious-flow detection; attacker purpose must be distinguished.
[32] Hofer et al. 2026 • P	T Injection	AgentDojo subset; 80 task pairs and automated attack optimization.	Black-box search outperforms the tested gradient-based adaptation; cross-family transfer remains limited.	Preprint; subset coverage, imperfect judging, and a single gradient-source model limit inference.
[33] Carlini et al. 2024	T Poisoning	Distributed web datasets; mutable-source and snapshot-timing attacks.	Dataset distribution infrastructure itself can create practical opportunities for contamination.	Demonstrated ingestion exposure does not establish compromise of every downstream trained model.
[34] Chen et al. 2023	S Assurance	Chronological Android malware detection;	Concept drift degrades detection; selective relabeling	Analyst labeling cost and trust in incoming samples matter when applying continual updates.

Study	Scope and theme	System and access	Evidence contribution	Limit on interpretation
		active and contrastive learning.	and retraining improve longitudinal performance.	
[35] Evertz et al. 2026	S Assurance	Methodological audit of 72 security and software-engineering papers.	Identifies nine recurring pitfalls and illustrates their consequences in empirical case studies.	The audit concerns research validity; it cannot supply a universal attack or defense effect size.

4. Threat Model and Conceptual Synthesis

A threat model should identify the protected asset the adversary's objective and entry point and the knowledge control and resources available to the attacker. Assets may include model integrity detection coverage sensitive telemetry software correctness and authority over infrastructure. Possible points of influence include training submissions client updates malware bytes packet sequences retrieved documents and tool responses. The terms white-box and black -box describe knowledge or interface access. They do not establish whether an attack preserves malicious functionality or can lead to an unauthorized action.

Figure 3 presents a lifecycle view of how failures may interact. Contamination can affect either a model or its retrieval store while manipulated observations can influence detection or an analyst's interpretation. The consequences depend on the next trust boundary. When an unverified interpretation can authorize a tool operation a wrong answer may lead to further harm. This sequence is an architectural inference the selected studies did not jointly measure it as a causal pathway.

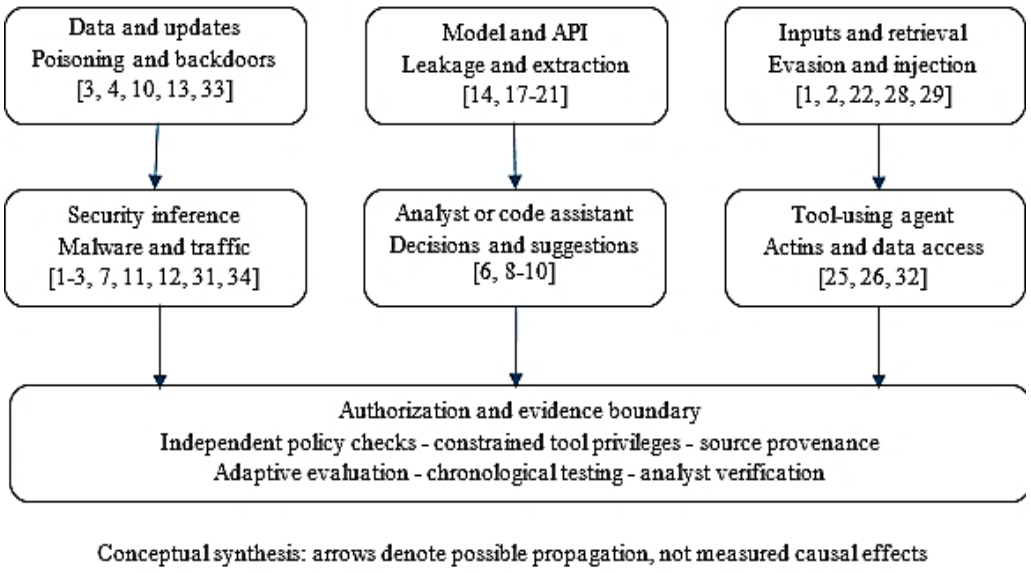


Figure (3): Attack surfaces and possible propagation through an AI security workflow. The control layer represents the assurance architecture proposed in this review and has not been validated as a universal defense

Table (3): Threat categories and distinctions needed to interpret their security consequences

Failure class	Attacker influence	Outcome that must be measured	Typical inference to avoid
Evasion	Executable artifacts, event sequences, or traffic at inference time.	Missed malicious activity under validated semantics and resource constraints.	A perturbed feature vector necessarily corresponds to a working attack.
Poisoning and backdoors	Training records, model updates, source repositories, or retrieval stores.	Targeted failure and benign utility after contamination.	A small nominal poison fraction means equal feasibility in every ingestion pipeline.
Privacy and extraction	Model queries, gradients, or attack-assisted training influence.	Membership, record reconstruction, text disclosure, or parameter recovery, separately.	Membership inference and full model cloning are interchangeable.
Jailbreak and injection	Direct adversarial requests or instructions embedded in untrusted content.	Disallowed content, task hijack, exfiltration, or unauthorized execution, separately.	An affirmative response proves executable compromise.
Drift and assurance	Changing distributions, contaminated evaluation, or human reliance.	Temporal detection quality, valid measurement, and downstream human decisions.	A benchmark score remains valid indefinitely or causes harm by itself.

5. Critical Analysis of Vulnerabilities

5.1 Evasion and the Validity of Attacker Manipulations:

Malware studies provide stronger evidence of evasion when the manipulated artifact retains its malicious function. Pierazzi et al. formalize the constraints linking feature changes to executable transformations [1], while Demetrio et al. demonstrate black-box evasion of Windows detectors by inserting benign content [2]. These findings show that static detectors can depend on evidence an attacker can alter without removing malicious functionality. Their scope remains limited: they do not demonstrate simultaneous bypass of an endpoint product's behavioral, signature, and containment components.

Network studies face comparable validity requirements. Nasr et al. manipulate live traffic without advance knowledge of the full connection [31]. PANTS combines optimization with explicit semantic constraints to generate realizable traffic for robustness assessment [11]. The contribution lies in showing that a manipulation is admissible, rather than merely increasing an attack-success score. Packet changes must respect constraints on direction, timing, representation, and attacker location. Purpose also matters: resistance to website fingerprinting can protect privacy, so evasion of a traffic classifier is not inherently malicious.

PROVNINJA addresses provenance-based detection by implementing evasive system actions rather than limiting the experiment to synthetic feature changes [7]. It illustrates the need to demonstrate both preservation of the malicious objective and failure of detection. In security operations, this suggests testing the full event pipeline including collection and feature extraction. Testing the

classifier alone cannot establish whether upstream normalization removes the manipulation or downstream correlation still generates an alert

Evaluating a defense introduces further requirements. Croce and Hein demonstrate the value of complementary attacks [16] and Tramèr et al. show why attacks must also be adapted to individual defenses [15]. A standard attack suite can therefore provide an initial diagnostic, followed by tests of defense-specific weaknesses. These findings do not support applying image-space norms directly to executable files. For an operational detector domain constraints should define the admissible manipulations before an optimizer is selected.

5.2 Poisoning Across Training and Retrieval:

Poisoning requires an identifiable path into training data or model updates. Severi et al. demonstrate clean-label backdoors in settings linked to malware feeds [3], and Schuster et al. show that contaminated code-completion training can induce insecure suggestions [4]. TrojanPuzzle further shows that screening for a complete suspicious payload may miss poison whose components are reconstructed during generation [10]. Taken together, these studies explain how apparently acceptable source material can produce harmful learned behavior. They establish limits on provenance checks and content scanning when used alone, rather than showing that either control has no value.

Federated model replacement [13] and contamination of web-scale datasets [33] rely on different access assumptions. The first allows a participant to manipulate a model update; the second exploits mutable distributed content during ingestion. Equal nominal poisoning percentages would not make these capabilities equivalent. Before applying a published contamination budget, an organization needs to determine who can submit examples, change labels, alter upstream repositories, or contribute updates to its own pipeline.

Retrieval systems permit poisoning without changes to model weights. AgentPoison targets persistent memory or knowledge stores [28], whereas PoisonedRAG targets evidence retrieved for question answering [29]. A threat-intelligence entry or a previous incident note could provide an analogous source for a security assistant, although this is an inference about transfer to that domain. Operational impact depends on the material being admitted, retrieved, trusted, and used in a consequential decision. Controls confined to the foundation model do not address all of these application conditions.

Truth Serum demonstrates a connection between integrity and confidentiality: adversarial influence over training can increase inference leakage [19]. For shared training and external contributions, protecting one participant's raw records does not establish safe behavior when another participant can choose inputs strategically. The mechanism motivates separate controls for ingestion integrity, model updates, and information released during inference. This recommendation is an architectural implication, not an experimentally established result for federated intrusion detection.

5.3 Leakage and Extraction Require Distinct Security Claims:

Gradient inversion [14] extraction of training text [17] membership inference [18] and partial model extraction [21] expose different assets through different interfaces. A single label such as "data leakage" obscures these differences. Inferring that a record was used in training does not recover its content, and recovering a projection layer does not clone a complete language model. Security assessments should identify the recovered asset and the access that enabled recovery.

Membership inference shows why average performance measures may be misleading. LiRA focuses on reliable inference at low false-positive rates [18]. When an organizational dataset has many possible nonmembers, high balanced accuracy may still provide weak evidence that a particular record was included. Evaluation should therefore report true-positive rates at false-positive rates relevant to the intended use with uncertainty estimates and a description of the candidate population. This is a recommendation about measurement, not a claim that every record faces the same privacy risk.

Kandpal et al. find that deduplication reduces exposure associated with memorization in the language models they examine [20]. This offers a practical data-engineering control although an observed reduction in leakage does not establish a universal privacy guarantee. Repeated incident templates, near duplicates and distinctive records may produce different exposure patterns. Their effects in cybersecurity require targeted tests using authorized data. Telemetry minimization and access controls remain necessary when training data have been deduplicated.

The production-model extraction study [21] shows that API design belongs in the threat model. Information returned by an interface can enable recovery that a different output policy would make more difficult. The result does not establish that suppressing scores alone provides sufficient protection. Deployment assessments should document outputs, logging query permissions, and monitoring then test the relevant leakage mechanism under those conditions.

5.4 Jailbreaking and Indirect Prompt Injection:

Jailbreaking and indirect prompt injection concern different boundaries. Adversarial suffixes [23] AutoDAN [24] and adaptive jailbreaks [30] test departures from intended model behavior under hostile prompts. Greshake et al. examine how content supplied by a third party can redirect an integrated application [22]. Analysts may have legitimate reasons to inspect hostile text. If an assistant treats that text as an instruction with authority, the inspection itself becomes a route for changing its behavior. A refusal benchmark does not adequately test this boundary.

Tensor Trust provides interpretable examples of attacks devised by people [27]. InjecAgent examines tool-integrated scenarios including exfiltration [26] while AgentDojo measures security alongside completion of legitimate tasks [25]. These evaluations make outcomes more relevant to application use. An assistant that rejects every input may appear secure while failing its purpose. Evaluation must

distinguish ordinary task failure authorized completion and behavior induced by an attacker and record whether malicious content reached the relevant tool-visible context

Adaptive attack results depend on the model interface prompt construction and evaluation procedure [30]. Success measured against a historical model version cannot be generalized to every current assistant. An affirmative opening or refusal keyword is also an inadequate sole indicator of security failure. For agents, evidence should establish the disclosure action or state change that violates the intended policy

In a subset of Agent Dojo, the 2026 preprint by Hofer et al. reports that the tested black-box optimization outperforms its gradient-based adaptation [32]. Limited transfer across model families cautions against assuming that a suffix successful on one model will compromise another. The acknowledged limits of the subset and automated judging also separate two sources of uncertainty, the quality of attack generation and the validity of outcome measurement. These provisional findings inform the threat model without establishing a universal ordering of attack families.

5.5 Insecure Code Generation and Human Reliance:

Audits of generated code and studies involving developers examine different stages of software production. Pearce et al. show that generated suggestions can contain vulnerabilities [6]. Perry et al. report less secure code and greater confidence in code security in their assistant condition [8]. Sandoval et al. find a comparatively small security impact in a specific low-level C exercise [9]. Differences in tasks, participant experience, assistant configuration, and opportunities to inspect or revise suggestions are therefore essential to interpreting these results.

The contrasting findings support conclusions tied to the tested conditions. A vulnerable model output, a user's acceptance of it subsequent testing and deployment are separate events. Security properties should be checked independently of the assistant's explanation. Effective human review requires relevant evidence a suitable review task and authority to intervene a nominal approval step alone does not demonstrate risk reduction.

5.6 Distribution Shift and Research Validity:

Chen et al. examine declining Android detection performance over time and an active-learning response [34]. Such deterioration can occur without deliberate manipulation by an adversary. Retraining may restore useful distinctions, but it also creates further opportunities to influence incoming samples and labels. Maintaining performance thus requires new evidence, while preserving integrity requires deciding whether that evidence is trustworthy.

Audits conducted by Arp et al. [5] and Evertz et al. [35] explain why apparently strong evaluations need methodological scrutiny. The 2026 audit identifies nine recurring pitfalls across 72 papers including problems affecting reproducibility and interpretation. These are findings about research practice not the prevalence of attacks. They support documenting model identity data provenance context handling and outcome definitions whenever evaluation is used to justify a security conclusion.

6. Defense Evidence and Deployment Implications

6.1 Quantitative Findings in Context:

Table 4 presents selected quantities reported by the source studies. They concern different tasks and do not estimate a shared effect. Benchmark size, historical extraction cost, model F1, and attack success each require their own interpretation. The values are therefore retained with their experimental context and are not combined into an aggregate vulnerability score.

Table (4): Selected quantitative findings as reported by the source studies. Values retain their experimental context and are neither pooled nor ranked

Study	Reported quantity	Experimental context	Interpretation boundary
[34] Chen et al.	F1 decreased from 0.99 to 0.76 after six months.	An Android classifier trained on one year of data and evaluated on newer samples.	Evidence of temporal degradation in that setting, not a universal six-month decay rate.
[29] Zou et al.	90% attack success with five malicious texts per target question.	Target-answer corruption in evaluated RAG systems with knowledge stores containing millions of texts.	Conditional on admission, retrieval, target questions, and tested models; no incident prevalence inference.
[28] Chen et al.	Average attack success ≥80%; benign-performance impact ≤1%; poison rate <0.1%.	Three evaluated agent applications; poisoned memory or knowledge base and optimized triggers.	These are source-reported bounds for the evaluated configurations, not SOC deployment rates.
[26] Zhan et al.	1,054 test cases; 17 user tools and 62 attacker tools.	InjecAgent benchmark coverage of indirect injection.	Coverage counts indicate test construction; they do not measure real-world likelihood.
[21] Carlini et al.	Under US$20 to extract the projection matrix of the tested Ada and Babbage models.	Historical API conditions in the published production-model study.	A partial-model result with historical cost assumptions; not full cloning or a current service assessment.
[35] Evertz et al.	72 reviewed papers and nine recurring methodological pitfalls.	An audit of selected 2023–2024 security and software-engineering papers.	Research-validity evidence; no direct estimate of deployed attack frequency.

6.2 What Current Defenses Establish:

Defenses should be evaluated at the boundary they are intended to protect. Semantics aware augmentation improves observed robustness in the evaluated setting [11]. Certification establishes a mathematical claim within a defined perturbation region [12] and deduplication reduces measured exposure from memorization [20]. These results do not establish resistance to every form of poisoning traffic manipulation prompt injection, or downstream misuse. Their relevance depends on whether the protected boundary corresponds to the control available to the attacker.

Formal certification and adversarial testing provide different kinds of evidence. A certificate establishes a statement under specified assumptions a successful red-team attack demonstrates failure under the attempted manipulation. An unsuccessful search offers limited reassurance if coverage or

resources are inadequate. Combining these approaches is useful when their respective scope is explicit. Certification for selected packet perturbations for example, can coexist with vulnerability to drift contaminated sources, or an unrelated authorization error.

Table (5): Evidence for defenses and its limits. Rows marked as synthesis describe proposals developed in this review rather than direct experimental findings

Control and evidence	Boundary addressed	Remaining risk	Required local validation
Semantics-aware training [11]	Realizable perturbations represented during robustification.	Unmodeled transformations, constraints, traffic domains, and resource costs.	Independent attacks under changed capabilities; clean utility and latency.
Certified robustness [12]	Specified additive and discrete traffic perturbations.	Attacks outside the certified region; collection or task mismatch.	Certificate coverage at useful accuracy, overhead, and local traffic semantics.
Deduplication [20]	Repeated training sequences that increase memorization exposure.	Unique or near-duplicate sensitive records and other leakage interfaces.	Targeted extraction tests; documented privacy unit if formal privacy is used.
Source and update provenance; synthesis from [3, 13, 33]	Entry of untrusted data, code, or updates.	Trusted contributors can be compromised; authentic content can still be malicious.	Immutable versions, access history, sample quarantine, and rollback exercises.
Retrieval controls; synthesis from [22, 28, 29]	Admission and use of untrusted retrieved material.	High-ranked malicious evidence or instructions can still reach a model.	Source-level trust, adversarial documents, and correct evidence attribution.
External authorization; synthesis from [25, 26, 32]	Conversion of model proposals into tool actions or disclosures.	Policy mistakes, overbroad permissions, or incorrect approval context.	Deny-by-default tool scopes, execution traces, and tested policy violations.
Temporal monitoring [34] and evaluation discipline [5, 35]	Loss of validity through drift or research artifacts.	Monitoring can miss changes; metrics may misrepresent operational costs.	Chronological holdouts, model versions, adjudicated labels, and incident review.

6.3 Proposed Assurance Architecture:

The proposed architecture treats the AI component as a source of evidence. The architecture keeps controls that match the consequences of the AI component's output. During ingestion the architecture keeps source identities and immutable versions. The architecture also separates third-party assertions, from analyst-approved labels. Finally, the architecture makes quarantine. Rollback available. During inference observations should retain their provenance and retrieved instructions should not acquire authorization authority. At execution, policy checks should operate outside the generative model, with tools restricted to resources required by the approved task. These recommendations have not been evaluated together as a single system in this review.

Failure investigation requires records of model and prompt versions, retrieved-source identifiers, input transformations, proposed tool calls authorization decisions and resulting state changes. These records may themselves contain confidential information. Retention and access policies therefore require the same asset-specific assessment applied to model outputs. Before collecting prompts or

telemetry without restriction, organizations should determine what evidence they need to reconstruct an incident.

The required assurance should reflect the consequences of error. Read-only summarization alert prioritization code change generation and infrastructure command execution entail different levels of risk. Performance adequate for summarization may be insufficient for command execution. Deployment should begin by establishing useful performance on authorized tasks and characterizing adversarial failures. Action should then be enabled only where an independent policy boundary can contain the consequences.

7. Research Agenda and Reporting Requirements

There is a gap between showing how an attack works and proving that it actually causes real security problems in real-world situations. Research that focuses on the attack mechanism is still useful for understanding what goes wrong. To really understand the impact, we need studies that look at how these attacks work in actual systems. That means looking at how security tools gather data how analysts make decisions what permissions are in place and how recovery is handled. Claims of greater realism should be supported by measurement of these additional stages.

Table (6): Research priorities and minimum reporting requirements derived from the narrative synthesis

Priority	Research design	Minimum outcomes to report
Realizable security attacks	Use executable malware, valid event sequences, or live/replayed protocol-valid traffic.	Goal preservation, validity checks, detector outcome, resource overhead, and invalid-attempt count.
Adaptive and compositional defenses	Compare independent attack families and adapt attacks to each defense and interface.	Attack knowledge, budget, restarts, failures, benign utility, and complete pipeline behavior.
Operational agent security	Test malicious external evidence with explicit tool scopes and executable policy checks.	Authorized task completion, policy violations, data disclosure, reachability, and resulting state.
Longitudinal detection	Use chronological evaluation and separate update selection from held-out testing.	Per-period recall and false-positive rates, labeling effort, drift signals, and update provenance.
Privacy by asset	Distinguish training membership, content recovery, gradients, and parameter extraction.	Threat access, candidate population, low-FPR performance, leakage unit, and mitigation cost.
Human and organizational outcomes	Compare assistant conditions across tasks and expertise with security-aware evaluation.	Security defects, task correctness, editing behavior, confidence calibration, and review effort.
Reproducible reporting	Freeze data, model IDs, prompts, retrieval configurations, and scoring rules.	Artifact availability, seeds, uncertainty, version dates, unavailable details, and negative results.

Where practical outcome definitions should be preregistered and reported for unsuccessful as well as successful tests. Detector evaluations should measure false positives and false negatives at thresholds consistent with analyst capacity and expected event base rates. Agent studies should identify policy violations from execution traces while privacy studies should define success by the asset recovered. Repeated trials and confidence intervals are needed when stochastic generation or randomized training materially influence outcomes.

Independent replication should examine whether improvements persist under changed conditions. Relevant tests include later data collection a new source another model family a different query budget or stricter permissions. A defense effective only against its authors' attack implementation may still offer scientific insight but it gives operators limited assurance against adaptive adversaries. Reporting unsuccessful attacks and failures to transfer can help distinguish general mechanisms from implementation-specific effects.

8. Limitations of this Review

The review is limited by the prescribed corpus size and the purposive selection of studies. The 35 studies cannot capture the body of literature published between 2020 and 2026. No claim is made that the sample of 35 studies is statistically representative or that 35 studies have reached saturation. Limiting inclusion to English language research with primary sources favors more visible work while excluding some industrial evidence non-English publications and unpublished failures. The emphasis on security and machine learning conferences may also leave deployment reports and independent incident investigations underrepresented.

The breadth of bibliographic verification exceeds the depth of experimental extraction. All 35 entries were checked against primary records or author versions but analysis relied on abstracts and selected full texts rather than uniform independent double extraction. Fine-grained comparisons of datasets threat-model parameters numerical uncertainty and reproducibility are consequently limited. The review therefore supplies neither numerical quality scores for individual studies nor rankings of attack success across papers. Its statements of limitations are interpretive judgments not a validated risk-of-bias assessment.

The 2026 evidence consists of one peer-reviewed methodological audit and one preprint and does not comprehensively cover that year's research. The two retained preprints are distinguished from reviewed publications. General-purpose vision, language-model and agent studies explain mechanisms relevant to security architectures but their applicability to security operations center (SOC) workloads remains an explicit inference. Both the proposed architecture and the research priorities require empirical validation in those settings.

API behavior and model versions may change after a study is published. The review describes historical findings under the cited conditions, rather than providing a current vulnerability assessment of any named commercial service. It does not independently reproduce attacks or test a new defense. Its contribution is a transparent critical organization of selected evidence that can support further specialist assessment and revision for a target journal.

9. Conclusion

The selected studies locate AI security failures at several boundaries: observations that can be manipulated training or retrieval sources that can be contaminated model interfaces that reveal information and the conversion of outputs into human decisions or tool actions. Interpretation requires

a defined attacker model and an outcome measure suited to the protected asset. Realizable attacks adaptive evaluation, and testing over time strengthen the evidence whereas heterogeneous benchmarks cannot establish a universal vulnerability rate. Deployment assessment should combine model testing with provenance controls restricted authority and independent verification of consequential actions. These measures need evaluation within complete security workflows and each claim should remain confined to the conditions supported by its evidence.

References

Numbered reference style. Primary publisher or proceedings URLs are used when no DOI is supplied. References [23] and [32] are explicitly identified preprints. Bibliographic records were checked on 1 September 2026 publication years refer to the cited versions.

1. F. Pierazzi, F. Pendlebury, J. Cortellazzi, and L. Cavallaro, “Intriguing Properties of Adversarial ML Attacks in the Problem Space,” 2020 IEEE Symposium on Security and Privacy, IEEE, 2020. https://doi.org/10.1109/SP40000.2020.00073.
2. L. Demetrio, B. Biggio, G. Lagorio, F. Roli, and A. Armando, “Functionality-Preserving Black-Box Optimization of Adversarial Windows Malware,” IEEE Transactions on Information Forensics and Security, vol. 16, pp. 3469–3478, 2021. https://doi.org/10.1109/TIFS.2021.3082330.
3. G. Severi, J. Meyer, S. Coull, and A. Oprea, “Explanation-Guided Backdoor Poisoning Attacks Against Malware Classifiers,” 30th USENIX Security Symposium, pp. 1487–1504, 2021. https://www.usenix.org/conference/usenixsecurity21/presentation/severi.
4. R. Schuster, C. Song, E. Tromer, and V. Shmatikov, “You Autocomplete Me: Poisoning Vulnerabilities in Neural Code Completion,” 30th USENIX Security Symposium, pp. 1559–1575, 2021. https://www.usenix.org/conference/usenixsecurity21/presentation/schuster.
5. D. Arp, E. Quiring, F. Pendlebury, A. Warnecke, F. Pierazzi, C. Wressnegger, L. Cavallaro, and K. Rieck, “Dos and Don’ts of Machine Learning in Computer Security,” 31st USENIX Security Symposium, pp. 3971–3988, 2022. https://www.usenix.org/conference/usenixsecurity22/presentation/arp.
6. H. Pearce, B. Ahmad, B. Tan, B. Dolan-Gavitt, and R. Karri, “Asleep at the Keyboard? Assessing the Security of GitHub Copilot’s Code Contributions,” 2022 IEEE Symposium on Security and Privacy, pp. 754–768, 2022. https://doi.org/10.1109/SP46214.2022.9833571.
7. K. Mukherjee, J. Wiedemeier, T. Wang, J. Wei, F. Chen, M. Kim, M. Kantarcioglu, and K. Jee, “Evading Provenance-Based ML Detectors with Adversarial System Actions,” 32nd USENIX Security Symposium, pp. 1199–1216, 2023. https://www.usenix.org/conference/usenixsecurity23/presentation/mukherjee.

8. N. Perry, M. Srivastava, D. Kumar, and D. Boneh, "Do Users Write More Insecure Code with AI Assistants?," Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security, pp. 2785–2799, 2023. https://doi.org/10.1145/3576915.3623157.
9. G. Sandoval, H. Pearce, T. Nys, R. Karri, S. Garg, and B. Dolan-Gavitt, "Lost at C: A User Study on the Security Implications of Large Language Model Code Assistants," 32nd USENIX Security Symposium, pp. 2205–2222, 2023. https://www.usenix.org/conference/usenixsecurity23/presentation/sandoval.
10. H. Aghakhani, W. Dai, A. Manoel, X. Fernandes, A. Kharkar, C. Kruegel, G. Vigna, D. Evans, B. Zorn, and R. Sim, "TrojanPuzzle: Covertly Poisoning Code-Suggestion Models," 2024 IEEE Symposium on Security and Privacy, IEEE, 2024. https://doi.org/10.1109/SP54263.2024.00140.
11. M. Jin and M. Apostolaki, "Robustifying ML-powered Network Classifiers with PANTS," 34th USENIX Security Symposium, pp. 7291–7310, 2025. https://www.usenix.org/conference/usenixsecurity25/presentation/jin-minhao.
12. J. Yan, Z. Liu, Y. Xie, S. Liang, L. Liu, and K. Xu, "CertTA: Certified Robustness Made Practical for Learning-Based Traffic Analysis," 34th USENIX Security Symposium, pp. 7349–7368, 2025. https://www.usenix.org/conference/usenixsecurity25/presentation/yan-jinzhu.
13. E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to Backdoor Federated Learning," Proceedings of AISTATS, Proceedings of Machine Learning Research, vol. 108, pp. 2938–2948, 2020. https://proceedings.mlr.press/v108/bagdasaryan20a.html.
14. J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting Gradients—How easy is it to break privacy in federated learning?," Advances in Neural Information Processing Systems, vol. 33, NeurIPS 2020, 2020. https://proceedings.neurips.cc/paper/2020/hash/c4ede56bbd98819ae6112b20ac6bf145-Abstract.html.
15. F. Tramèr, N. Carlini, W. Brendel, and A. Madry, "On Adaptive Attacks to Adversarial Example Defenses," Advances in Neural Information Processing Systems, vol. 33, NeurIPS 2020, 2020. https://proceedings.neurips.cc/paper_files/paper/2020/hash/11f38f8ecd71867b42433548d1078e38-Abstract.html.
16. F. Croce and M. Hein, "Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks," Proceedings of ICML, Proceedings of Machine Learning Research, vol. 119, pp. 2206–2216, 2020. https://proceedings.mlr.press/v119/croce20b.html.
17. N. Carlini, F. Tramèr, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. Brown, D. Song, Ú. Erlingsson, A. Oprea, and C. Raffel, "Extracting Training Data from Large Language Models," 30th USENIX Security Symposium, pp. 2633–2650, 2021. https://www.usenix.org/conference/usenixsecurity21/presentation/carlini-extracting.

18. N. Carlini, S. Chien, M. Nasr, S. Song, A. Terzis, and F. Tramèr, "Membership Inference Attacks From First Principles," 2022 IEEE Symposium on Security and Privacy, pp. 1897–1914, 2022. https://doi.org/10.1109/SP46214.2022.9833649.
19. F. Tramèr, R. Shokri, A. San Joaquin, H. Le, M. Jagielski, S. Hong, and N. Carlini, "Truth Serum: Poisoning Machine Learning Models to Reveal Their Secrets," Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, pp. 2779–2792, 2022. https://doi.org/10.1145/3548606.3560554.
20. N. Kandpal, E. Wallace, and C. Raffel, "Deduplicating Training Data Mitigates Privacy Risks in Language Models," Proceedings of ICML, Proceedings of Machine Learning Research, vol. 162, pp. 10697–10707, 2022. https://proceedings.mlr.press/v162/kandpal22a.html.
21. N. Carlini, D. Paleka, K. D. Dvijotham, T. Steinke, J. Hayase, A. F. Cooper, K. Lee, M. Jagielski, M. Nasr, A. Conmy, E. Wallace, D. Rolnick, and F. Tramèr, "Stealing part of a production language model," Proceedings of ICML, Proceedings of Machine Learning Research, vol. 235, pp. 5680–5705, 2024. https://proceedings.mlr.press/v235/carlini24a.html.
22. K. Greshake, S. Abdelnabi, S. Mishra, C. Endres, T. Holz, and M. Fritz, "Not What You've Signed Up For: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection," Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security, AISec 2023, 2023. https://doi.org/10.1145/3605764.3623985.
23. A. Zou, Z. Wang, J. Z. Kolter, and M. Fredrikson, "Universal and Transferable Adversarial Attacks on Aligned Language Models," arXiv:2307.15043v1, July 27, 2023, preprint, 2023. https://arxiv.org/abs/2307.15043v1.
24. X. Liu, N. Xu, M. Chen, and C. Xiao, "AutoDAN: Generating Stealthy Jailbreak Prompts on Aligned Large Language Models," International Conference on Learning Representations, ICLR 2024, 2024. https://openreview.net/forum?id=7Jwpw4qKkb.
25. E. Debenedetti, J. Zhang, M. Balunović, L. Beurer-Kellner, M. Fischer, and F. Tramèr, "AgentDojo: A Dynamic Environment to Evaluate Prompt Injection Attacks and Defenses for LLM Agents," Advances in Neural Information Processing Systems, vol. 37, Datasets and Benchmarks Track, 2024, 2024. https://doi.org/10.52202/079017-2636.
26. Q. Zhan, Z. Liang, Z. Ying, and D. Kang, "InjecAgent: Benchmarking Indirect Prompt Injections in Tool-Integrated Large Language Model Agents," Findings of the Association for Computational Linguistics: ACL 2024, pp. 10471–10506, 2024. https://doi.org/10.18653/v1/2024.findings-acl.624.
27. S. Toyer, O. Watkins, E. Mendes, J. Svegliato, L. Bailey, T. Wang, I. Ong, K. Elmaaroufi, P. Abbeel, T. Darrell, A. Ritter, and S. Russell, "Tensor Trust: Interpretable Prompt Injection Attacks from an Online Game," International Conference on Learning Representations, ICLR 2024, 2024. https://proceedings.iclr.cc/paper_files/paper/2024/hash/519c51529c3544b3430bd8b17d400365-Abstract-Conference.html.

28. Z. Chen, Z. Xiang, C. Xiao, D. Song, and B. Li, "AgentPoison: Red-teaming LLM Agents via Poisoning Memory or Knowledge Bases," Advances in Neural Information Processing Systems, vol. 37, NeurIPS 2024, 2024. https://doi.org/10.52202/079017-4136.
29. W. Zou, R. Geng, B. Wang, and J. Jia, "PoisonedRAG: Knowledge Corruption Attacks to Retrieval-Augmented Generation of Large Language Models," 34th USENIX Security Symposium, pp. 3827–3844, 2025. https://www.usenix.org/conference/usenixsecurity25/presentation/zou-poisonedrag.
30. M. Andriushchenko, F. Croce, and N. Flammarion, "Jailbreaking Leading Safety-Aligned LLMs with Simple Adaptive Attacks," International Conference on Learning Representations, ICLR 2025, 2025. https://proceedings.iclr.cc/paper_files/paper/2025/hash/63fa7efdd3bcf944a4bd6e0ff6a50041-Abstract-Conference.html.
31. M. Nasr, A. Bahramali, and A. Houmansadr, "Defeating DNN-Based Traffic Analysis Systems in Real-Time With Blind Adversarial Perturbations," 30th USENIX Security Symposium, pp. 2705–2722, 2021. https://www.usenix.org/conference/usenixsecurity21/presentation/nasr.
32. D. Hofer, E. Debenedetti, and F. Tramèr, "Assessing Automated Prompt Injection Attacks in Agentic Environments," arXiv:2606.10525v1, June 9, 2026, preprint, 2026. https://doi.org/10.48550/arXiv.2606.10525.
33. N. Carlini, M. Jagielski, C. A. Choquette-Choo, D. Paleka, W. Pearce, H. Anderson, A. Terzis, K. Thomas, and F. Tramèr, "Poisoning Web-Scale Training Datasets is Practical," 2024 IEEE Symposium on Security and Privacy, pp. 407–425, 2024. https://doi.org/10.1109/SP54263.2024.00179.
34. Y. Chen, Z. Ding, and D. Wagner, "Continuous Learning for Android Malware Detection," 32nd USENIX Security Symposium, pp. 1127–1144, 2023. https://www.usenix.org/conference/usenixsecurity23/presentation/chen-yizheng.
35. J. Evertz, N. Risse, N. Neuer, A. Müller, P. Normann, G. Sapia, S. Gupta, D. Pape, S. Shaw, D. Srivastav, C. Wressnegger, E. Quiring, T. Eisenhofer, D. Arp, and L. Schönherr, "Chasing Shadows: Pitfalls in LLM Security Research," Network and Distributed System Security Symposium, NDSS 2026, Internet Society, 2026. https://www.ndss-symposium.