Tracing the Adoption of Adversarial Machine Learning Research into Industry Practice (2014–2025)

[Author Names] Affiliations>Affiliations

[emails]

Abstract

Despite a decade of adversarial machine learning (AML) research producing over 66 catalogued attack techniques and numerous defense mechanisms, industry adoption remains limited—surveys indicate only 5% of practitioners have experienced AI-specific attacks, yet 86% express security concerns. This systematic review measures the temporal lag between publication of landmark AML research and evidence of industry adoption across tool integration, commercial deployment, regulatory citation, and production use. We analyze approximately 120 papers published between 2014–2025, spanning foundational attacks (FGSM, C&W, PGD), privacy threats (membership inference, model extraction), physical-world attacks, and LLM-specific vulnerabilities (jailbreaking, prompt injection). Our findings reveal domain-dependent adoption lags ranging from 4–6 years for malware detection to 10+ years for financial systems, with LLM security exhibiting compressed 1–2 year cycles driven by direct user interaction and regulatory pressure. We identify key acceleration factors including standardized evaluation (AutoAttack, RobustBench, HarmBench), regulatory mandates (EU AI Act Article 15, NIST AI RMF), and significant commercial investment in AI security. Our coding framework and adoption evidence database provide a foundation for future research-practice alignment studies.

1 Introduction

Machine learning systems now influence decisions affecting millions daily—from facial recognition at border crossings (**?**) to fraud detection in financial services (**?**) and content moderation on social platforms (**?**). This widespread deployment has intensified scrutiny of adversarial vulnerabilities: inputs or interactions crafted to cause models to behave in unintended ways (**?**).

The field of adversarial machine learning emerged with Szegedy et al.'s demonstration that imperceptible perturbations could fool state-of-the-art classifiers (**?**), followed by Goodfellow et al.'s Fast Gradient Sign Method establishing the dominant attack paradigm (**?**). Over the subsequent decade, researchers catalogued 66 attack techniques spanning evasion, poisoning, and privacy threats (**?**). MITRE ATLAS now documents 33 real-world case studies, and regulatory frameworks including the EU AI Act mandate adversarial robustness testing for high-risk systems (**?**).

Yet industry surveys reveal a persistent gap. Kumar et al. (**?**) found practitioners "not equipped with tactical and strategic tools" for ML-specific attacks. Grosse et al. (**?**) reported only 5% of AI practitioners had experienced AI-specific attacks, despite 86% expressing concern. Mink et al. (**?**) identified organizational barriers including lack of institutional motivation, inability to assess AML risk, and structures discouraging implementation. Apruzzese et al. (**?**) crystallized these concerns, arguing that "real attackers don't compute gradients"—academic threat models assume capabilities rarely available in practice.

This gap matters because real-world incidents demonstrate adversarial threats are not merely theoretical. In November 2025, Anthropic disclosed the first documented large-scale AI-orchestrated cyber cam-

paign, with Chinese state-sponsored actors using Claude Code to execute 80–90% of operational tasks autonomously (**?**). Tesla Autopilot was fooled by adversarial tape modifications (**??**). Training data extraction from ChatGPT recovered megabytes of verbatim training data for under $200 (**?**). The OWASP Top 10 for LLM Applications lists prompt injection as the #1 vulnerability across all versions (**?**).

1.1 Research Questions

We address three questions about the research-to-practice transfer in adversarial ML:

1. **RQ1 (Adoption Lag):** What is the typical time lag between publication of landmark AML research and evidence of industry adoption, measured through tool integration, commercial reference, regulatory citation, and production deployment?
2. **RQ2 (Domain Variation):** How does adoption speed vary across application domains (computer vision, NLP, malware detection, autonomous systems, LLMs), and what factors explain these differences?
3. **RQ3 (Acceleration Factors):** What mechanisms—regulatory frameworks, standardized benchmarks, industry consortiums, commercial tools—have accelerated adoption, particularly for foundation model security post-2022?

1.2 Contributions

This review makes four contributions:

1. **Artifact-anchored methodology:** We introduce a reverse-engineering approach that traces industry artifacts back to source papers, ensuring every paper in our sample has verified adoption evidence by construction.
2. **Quantified adoption timelines:** We provide the first systematic measurement of research-to-industry lag across approximately 120 papers, documenting specific dates for tool integration, benchmark inclusion, regulatory citation, and vendor acknowledgment.
3. **Reproducible coding framework:** We release a 14-variable codebook (Table **??**) with explicit decision rules, enabling replication and extension to future papers.
4. **Domain-stratified analysis:** We document domain-specific patterns and identify factors (code availability, threat model realism, regulatory pressure) that predict faster adoption.

2 Background

2.1 The Emergence of Adversarial Vulnerabilities

Modern adversarial ML research began with Szegedy et al.'s December 2013 demonstration that small perturbations could cause misclassification with high confidence (**?**). This work, which received ICLR's 2024 Test of Time Award, revealed that adversarial examples transfer across independently trained models—a finding with profound security implications.

Goodfellow et al. (**?**) attributed these vulnerabilities to linear behavior in high-dimensional spaces and introduced the Fast Gradient Sign Method (FGSM), appearing on arXiv December 20, 2014 and published at ICLR 2015. FGSM established gradient-based perturbation as the dominant paradigm and introduced adversarial training as a defense. These foundational works established conventions that shaped subsequent research: white-box access assumptions, L_p perturbation constraints, and optimization-based attack formulations.

Subsequent work refined attack formulations. The Carlini & Wagner attack (**?**) (IEEE S&P 2017) achieved state-of-the-art success across multiple norms. Madry et al.'s PGD attack (**?**) (ICLR 2018) estab-

lished the robust optimization framework:

$$\min_{\theta} \mathbb{E}_{(x,y)\sim D} \left[\max_{\|\delta\| \leq \epsilon} L(f_\theta(x+\delta), y) \right] \tag{1}$$

This min-max formulation remains the gold standard—10 of the top-10 models on RobustBench use PGD-based adversarial training (**?**).

2.2 Expanding Threat Landscape

Research expanded beyond test-time evasion to encompass the full ML pipeline. **Privacy attacks** demonstrated that models leak information: Shokri et al. (**?**) introduced membership inference at IEEE S&P 2017; Tramèr et al. (**?**) demonstrated model extraction from commercial APIs at USENIX Security 2016; Carlini et al. (**?**) extracted verbatim training data from GPT-2 at USENIX Security 2021.

Integrity attacks compromise models during training. BadNets (**?**) (arXiv August 2017) demonstrated backdoor injection through poisoned training data. Subsequent work extended backdoors to federated learning (**?**) and self-supervised learning (**?**).

Physical-world attacks demonstrated that adversarial examples survive real-world conditions. Eykholt et al. (**?**) (CVPR 2018) created adversarial stop signs robust to viewing angles and distances. DolphinAttack (**?**) (ACM CCS 2017, Best Paper) compromised voice assistants via ultrasonic commands inaudible to humans.

2.3 The LLM Security Paradigm Shift

Large language models introduced qualitatively different adversarial challenges. Unlike traditional attacks requiring gradient access and imperceptible perturbations, LLM attacks exploit semantic properties through natural language.

Prompt injection was first documented by Simon Willison in September 2022 and formalized by Perez & Ribeiro (**?**) at the NeurIPS 2022 Workshop on ML Safety. Greshake et al. (**?**) demonstrated indirect prompt injection against retrieval-augmented systems (arXiv February 2023, ACM AISec November 2023), showing attackers can embed malicious instructions in external content.

Automated jailbreaking emerged with Zou et al.'s GCG attack (**?**) (arXiv July 2023, NeurIPS 2023 Spotlight), which optimizes adversarial suffixes achieving the first automated jailbreaks against aligned LLMs including ChatGPT, Bard, and Claude. Subsequent work developed more efficient black-box methods: PAIR (**?**) achieves jailbreaks in approximately 20 queries; TAP (**?**) uses tree-of-thought reasoning for further efficiency.

LLM defenses include alignment techniques (RLHF (**?**), Constitutional AI (**?**)), specialized guardrails (LlamaGuard (**?**) released December 2023, NeMo Guardrails open-sourced April 2023), and input/output filtering. However, these defenses exhibit brittleness—the HackAPrompt competition (**?**) saw all 44 defenses eventually bypassed across 137,000+ adversarial interactions.

2.4 The Theory-Practice Gap

Apruzzese et al. (**?**) synthesized concerns about research-practice disconnect through real-world case studies at IEEE SaTML 2023. Their core observation: academic threat models assume attackers possess white-box access, unlimited queries, and gradient computation capabilities rarely available in practice. Real incidents typically involve simpler tactics—basic input manipulation, system-level exploitation, social engineering.

Industry surveys quantify this gap. Kumar et al. (**?**) interviewed 28 organizations at IEEE S&P Workshops 2020, finding widespread uncertainty about assessing adversarial risks. Grosse et al. (**?**) surveyed 139

practitioners (IEEE TIFS 2023), finding only 5% had experienced AI-specific attacks despite 86% expressing concern. Mink et al. (**?**) conducted 21 interviews at USENIX Security 2023, identifying three barriers: lack of institutional motivation, inability to assess AML risk, and organizational structures discouraging implementation.

However, prior work characterized this gap qualitatively. Our contribution is to measure adoption timelines quantitatively, identifying specific lag durations and acceleration factors. This work does not assess the severity of adversarial threats, but quantitatively measures when research techniques enter practitioner ecosystems.

3 Methodology

3.1 Study Design Overview

This study measures the temporal lag between adversarial machine learning (AML) research and demonstrable industry adoption. Rather than starting from academic publications and speculating about downstream impact, we adopt an *artifact-anchored backward traceability* methodology. We begin with concrete industry artifacts—open-source security tools, standardized benchmarks, regulatory frameworks, and vendor security documentation—and trace each adopted technique back to the academic paper that originally introduced it.

This design ensures that every paper in our dataset has verifiable evidence of adoption by construction, enabling precise measurement of research-to-practice timelines while avoiding subjective judgments about a paper's "importance" or speculative claims about industry relevance.

3.2 Artifact Universe Definition

Before extracting any papers, we define and freeze the set of industry artifacts examined in this study. Artifacts were selected according to objective inclusion criteria to minimize selection bias.

3.2.1 Open-Source Security Tools

We include open-source AML tools satisfying at least one of the following criteria:

- $\geq$1,000 GitHub stars, or
- $\geq$100 academic citations (via Semantic Scholar)

Based on these criteria, we analyze five tools:

1. **IBM Adversarial Robustness Toolbox (ART)** (**?**): 4,800+ stars, Linux Foundation AI project (graduated February 2022)
2. **CleverHans** (**?**): 6,100+ stars, first major AML library (October 2016)
3. **Foolbox**: 2,700+ stars, extensive attack implementations
4. **TextAttack**: 2,900+ stars, NLP-specific attacks
5. **Microsoft PyRIT** (**?**): LLM red-teaming toolkit (February 2024)

For each tool, we extract: (a) technique name, (b) source paper citation from documentation/docstrings, (c) date of first Git commit implementing the technique (via `git log -follow`), and (d) release version containing the technique.

3.2.2 Standardized Benchmarks

We include benchmarks that evaluate AML attacks or defenses as named techniques and are used in peer-reviewed research or cited by industry:

1. **RobustBench** (**?**): Leaderboard tracking 120+ defense models with paper citations. We record when each defense was added to the leaderboard (via GitHub commit history).

2. **AutoAttack** (**?**): Ensemble evaluation method with component attacks (APGD, FAB, Square Attack). We extract source papers for each component.
3. **HarmBench** (**?**): Standardized jailbreak evaluation across 33 LLMs, evaluating 18 red-teaming methods with source paper citations.

Only techniques explicitly evaluated and attributed to specific academic papers are included.

3.2.3 Regulatory and Threat Frameworks

We analyze the following frameworks:

1. **MITRE ATLAS** (**?**): 66 adversarial ML techniques with academic references. We extract technique ID, name, and cited papers. Technique page creation dates determined via Wayback Machine archives.
2. **NIST AI RMF Adversarial ML Taxonomy** (AI 100-2e2025): Technique categorization with literature references. Publication date: January 2023 (v1.0), updated July 2024 (GenAI Profile).
3. **OWASP Top 10 for LLM Applications** (**?**): Vulnerability descriptions citing foundational research. Version 1.0 (August 2023) and Version 2.0 (November 2024) analyzed.

We extract only techniques explicitly linked to academic references. Descriptive categories without citations are excluded.

3.2.4 Vendor Security Documentation

We analyze publicly available security documentation from major cloud and AI providers:

- AWS SageMaker security documentation
- Microsoft Azure AI security whitepapers and Responsible AI documentation
- Google Cloud Vertex AI security guides
- OpenAI system cards and security disclosures
- Anthropic system cards and security disclosures

Only documents that explicitly reference AML techniques or academic work are included. For each reference, we record: (a) technique name, (b) document URL, (c) document publication date, and (d) cited academic paper (if any).

3.3 Artifact-to-Paper Extraction

For each artifact, we extract tuples of the form:

$$(\text{artifact\_name}, \text{artifact\_type}, \text{technique\_name}, \text{cited\_reference}) \tag{2}$$

Extraction follows these rules:

1. The technique must be named explicitly (e.g., "FGSM," "Carlini–Wagner," "GCG").
2. The artifact must either cite a paper directly or attribute the technique to identifiable authors.
3. Techniques lacking an academic anchor are excluded.

This process yields a set of candidate academic papers that have demonstrably entered practitioner workflows.

3.4 Paper Canonicalization and Validation

Each extracted reference is resolved to a canonical paper record:

- **Identifier:** DOI where available, otherwise arXiv identifier

- **Publication date:** First arXiv submission date if it precedes peer-reviewed publication, otherwise conference/journal publication date. This reflects the earliest point at which the technique could plausibly be adopted.

Each paper is manually validated to ensure:

1. The paper introduces the technique (not merely applies or evaluates it)
2. The technique name corresponds to the extracted artifact reference
3. The paper is publicly accessible

Papers failing any validation criterion are excluded.

3.5 Adoption Event Definition

We define four observable adoption events, each treated independently:

Table 1: Adoption event definitions

Adoption Event	Operational Definition
Tool adoption	First Git commit implementing the technique
Benchmark adoption	Technique evaluated as a named method
Regulatory adoption	Technique referenced in regulatory framework
Vendor acknowledgment	Technique referenced in vendor security documentation

Adoption is defined strictly as acknowledgment or integration, not mitigation or successful defense.

3.6 Adoption Lag Measurement

For each (paper, adoption event) pair, we compute:

$$\text{Adoption Lag} = \text{Date}_{\text{artifact}} - \text{Date}_{\text{publication}} \tag{3}$$

Artifact dates are defined as:

- **Tools:** Git commit timestamp (UTC)
- **Benchmarks:** Leaderboard inclusion date or paper publication date
- **Regulatory:** Framework publication or technique page creation date (Wayback Machine)
- **Vendor:** Document publication date (from document metadata or press release)

Publication dates are defined as:

- **Peer-reviewed:** Conference/journal publication date
- **arXiv-first:** First arXiv submission date

For papers appearing in multiple artifacts, we report: (a) *first adoption lag* (minimum across all events), (b) *median adoption lag*, and (c) *regulatory adoption lag* (time to MITRE/NIST/OWASP inclusion, if any).

3.7 Paper Coding Scheme

Each paper is coded along three dimensions using a fixed codebook (Table **??**). All coding is performed by the authors based on the paper text only. Threat model attributes are coded using author-stated assumptions; if unstated, code as "Not specified."

Table 2: Complete codebook for paper coding

Variable	Values	Coding Rule
Research Characteristics (G1–G7)		
G1: Technique type	Attack / Defense / Evaluation	Primary contribution of paper
G2: Threat category	Evasion / Poisoning / Privacy / Availability / Multiple	Attack type(s) addressed
G3: Domain	Vision / NLP / Malware / Audio / Tabular / LLM / Cross-domain	Primary evaluation domain
G4: Venue type	ML / Security / Journal / arXiv-only	Publication venue category
G5: Code available	Yes / No	Code link in paper or GitHub
G6: Code timing	At-publication / Post / Never	When code was released
G7: Year	2014–2025	First public availability (arXiv or venue)
Threat Model Assumptions (T1–T4)		
T1: Access level	White-box / Gray-box / Black-box	White = full access; Gray = surrogate/partial; Black = query-only
T2: Gradient required	Yes / No	Any gradient computation required
T3: Query budget	Unlimited / >1000 / 100–1000 / <100 / Zero	Stated or implied query count
T4: Perturbation	L_∞ / L_2 / L_0 / Semantic / Unconstrained	Primary constraint used
Practical Evaluation (Q1–Q3)		
Q1: Real-world eval	Yes / Partial / No	Yes = deployed system tested
Q2: Cost reported	Yes / Qualitative / No	Yes = explicit metrics (FLOPs, time, queries)
Q3: Defense-aware	Yes / Partial / No / N/A	Adaptive attacks against defenses

3.7.1 Coding Decision Rules

To ensure consistency, we apply the following decision rules:

1. **G1 (Technique type):** If paper proposes both attack and defense, code based on which receives more experimental evaluation.
2. **G3 (Domain):** Code "Cross-domain" only if paper explicitly evaluates on 2+ distinct domains with

separate experiments.

3. **T1 (Access level):** If attack uses surrogate model for gradient computation then transfers to target, code as "Gray-box." If attack queries target model API without internal access, code as "Black-box."
4. **T2 (Gradient required):** Code "Yes" if gradients are used at *any* stage, including surrogate training. For LLM attacks using only text prompts, code "No."
5. **Q1 (Real-world evaluation):** Code "Yes" only if paper tests on a production/deployed system (e.g., commercial API, deployed autonomous vehicle). Code "Partial" for realistic simulations or industry-standard datasets designed to mimic deployment.
6. **Q3 (Defense-aware):** For defense papers, code "N/A." For attack papers, code "Yes" if paper explicitly tests against adaptive defenses or uses AutoAttack-style evaluation.

3.8 Coding Procedure

For each paper in the sample, the coder follows this step-by-step procedure:

1. **Record metadata:** Paper title, authors, DOI/arXiv ID, publication date (earliest of arXiv or venue), venue name.
2. **Read abstract and introduction:** Identify technique name, claimed contribution (attack/defense/evaluation), and stated domain.
3. **Locate threat model section:** Most papers have explicit "Threat Model" or "Adversary Capabilities" section. Extract access level, gradient requirements, query budget, and perturbation constraints.
4. **Review experiments:** Identify evaluation datasets, whether real-world/deployed systems are tested, and whether computational costs are reported.
5. **Check code availability:** Search paper for GitHub/code links. If not in paper, search GitHub for "[technique name] [first author name]."
6. **Code all variables:** Apply codebook (Table **??**) using decision rules. If uncertain, flag for second-coder review.
7. **Record adoption events:** For each artifact where the paper appears, record: artifact name, artifact type, adoption date (Git commit/document date).
8. **Calculate adoption lags:** Compute lag in months from publication date to each adoption event.

Estimated time per paper: 15–25 minutes for coding; additional 10–15 minutes for adoption event verification.

3.9 Statistical Analysis Plan

We address each research question through pre-specified analyses:

RQ1 (Adoption Lag): We report distributions of first-adoption lag and median-adoption lag, stratified by artifact type (tool, benchmark, regulatory, vendor). We compare lags across publication eras (2014–2017, 2018–2021, 2022–2025) using Kruskal-Wallis tests with post-hoc Dunn tests.

RQ2 (Domain Variation): We compare adoption lags across domains using pairwise Mann-Whitney U tests with Bonferroni correction ($\alpha = 0.05/\binom{n}{2}$ for n domains). Primary hypothesis: LLM security papers show significantly shorter lags than computer vision papers.

RQ3 (Acceleration Factors): We fit a Cox proportional hazards model predicting time-to-first-adoption, with covariates: publication year (continuous), domain (categorical), venue type (ML vs. Security), code availability (binary), and threat model (white-box vs. gray-box vs. black-box). Hazard ratios >1 indicate faster adoption. Model diagnostics include proportional hazards tests and residual analysis.

3.10 Reliability and Reproducibility

We employ three reliability checks:

1. **Intra-rater reliability:** 15% of papers (randomly selected, stratified by domain) are re-coded by the same coder after a two-week interval. We report Cohen's κ for each categorical variable and mean absolute deviation for adoption lag measurements.
2. **Inter-rater reliability:** For papers coded by multiple authors, we report pairwise κ and resolve disagreements through discussion until consensus. If no consensus, senior author decides.
3. **Adoption date verification:** All Git commit dates are verified independently using `git log`. Regulatory dates are verified against Wayback Machine archives. Discrepancies >30 days are flagged for review.

Target reliability: $\kappa \geq 0.80$ for all categorical variables; mean absolute deviation ≤ 2 months for adoption lag measurements.

Data release: All extracted metadata, coding decisions, raw Git commit hashes, and artifact URLs will be released as a structured CSV dataset with accompanying codebook. Code for adoption lag computation and statistical analyses will be released as a Python notebook.

3.11 Methodological Limitations

This methodology has several limitations:

1. **Selection bias toward adopted work:** By construction, our sample excludes papers that were never adopted. We cannot estimate the "adoption rate" (fraction of papers eventually adopted), only the adoption timeline for papers that were adopted.
2. **Tool coverage:** We analyze five major open-source tools, but techniques may be implemented in proprietary systems we cannot observe. Our timelines represent lower bounds on true first-adoption dates.
3. **Artifact dating precision:** Git commit dates are reliable, but vendor documentation dates may reflect updates rather than first publication. We use Wayback Machine archives where available to verify initial publication dates.
4. **Causal inference:** Observing that black-box attacks have shorter adoption lags does not prove that threat model choice *causes* faster adoption; both may be driven by domain characteristics or other confounds.
5. **Geographic and linguistic scope:** We focus on English-language publications and Western regulatory frameworks (US, EU). Adoption patterns in other regions may differ.

4 Industry Tools and Framework Timeline

The development of industry tools provides a concrete timeline for measuring research-to-practice transfer. Table **??** documents major tools with release dates verified from GitHub releases, arXiv papers, and official announcements.

Table 3: Major AML tools and release timeline

Tool	Release	Key Milestone
CleverHans	Oct 2016	arXiv:1610.00768
Foolbox	Jul 2017	ICML 2017 Workshop
IBM ART	Jul 2018	LF AI Feb 2022
TextAttack	May 2020	EMNLP 2020 Demo
Counterfit	May 2021	Microsoft Security
AutoAttack	Mar 2020	ICML 2020
RobustBench	Oct 2020	NeurIPS 2021 D&B
PyRIT	Feb 2024	LLM red-teaming
HarmBench	Feb 2024	ICML 2024

CleverHans (**?**) was created by Nicolas Papernot and Ian Goodfellow, implementing FGSM within two years of its publication. **IBM's Adversarial Robustness Toolbox** (**?**) released its first paper July 2018, was donated to Linux Foundation AI in July 2020, and graduated to full project status February 2022. **AutoAttack** (**?**) became the de facto evaluation standard after demonstrating that reported robust accuracies dropped by $>$10% for 13 published models when evaluated rigorously. **RobustBench** (**?**) now tracks 120+ models with standardized evaluation.

The LLM era introduced specialized tools. **Microsoft PyRIT** (**?**) (February 2024) focuses on LLM red-teaming. **HarmBench** (**?**) (February 2024) provides standardized jailbreak evaluation across 33 LLMs. Commercial tools followed: Azure Prompt Shields entered preview March 2024 and GA September 2024; Google Model Armor launched February 2025.

Regulatory frameworks codified these practices. **MITRE ATLAS** (**?**) launched June 2021, now cataloguing 66 techniques and 33 case studies. **NIST AI RMF 1.0** (**?**) was published January 26, 2023, with the Generative AI Profile following July 26, 2024. The **EU AI Act** (**?**) entered force August 1, 2024, with high-risk adversarial testing requirements effective August 2, 2026. **OWASP Top 10 for LLM Applications** (**?**) released v1.0 August 2023 and v2.0 November 2024.

5 Real-World Incidents

Documented incidents demonstrate that adversarial threats materialize in practice, though often through simpler mechanisms than academic threat models assume. We highlight three incidents with formal documentation in MITRE ATLAS, OWASP, or vendor disclosures:

Training data extraction from ChatGPT: Nasr, Carlini et al. (**?**) demonstrated that prompting ChatGPT to "repeat the word 'poem' forever" extracted several megabytes of training data for approximately $200, with $>$5% being verbatim 50-token copies. This attack exploited model memorization rather than adversarial optimization, requiring no gradient access or model knowledge.

Prompt injection vulnerabilities: CVE-2024-5184 (EmailGPT, CVSS 9.1) and CVE-2025-68664 (LangChain "LangGrinch," CVSS 9.3) document production prompt injection vulnerabilities (**?**). The Microsoft Bing Chat incident (February 2023) saw system prompt extraction within 24 hours of launch through simple conversational probing.

Agentic AI campaign: In November 2025, Anthropic disclosed the first documented large-scale AI-orchestrated cyber espionage operation (**?**). Chinese state-sponsored group GTG-1002 used Claude Code to execute 80–90% of operational tasks autonomously against approximately 30 targets. This incident exploited agentic capabilities and tool integration rather than traditional adversarial perturbations.

These incidents share a pattern: attackers exploit system integration points, deployment assumptions, and operational gaps rather than computing optimal L_p-bounded perturbations. This validates Apruzzese et

al.'s critique while demonstrating that adversarial vulnerabilities do manifest in practice.

6 Results

6.1 Sample Construction

Figure **??** illustrates the sample construction process following PRISMA guidelines adapted for artifact-anchored methodology.

Stage 1: Artifact Identification
- Tools: 5 repositories analyzed → [X] technique implementations
- Benchmarks: 3 benchmarks analyzed → [X] evaluated techniques
- Regulatory: 3 frameworks analyzed → [X] documented techniques
- Vendor: 5 vendors analyzed → [X] referenced techniques

Stage 2: Reference Extraction
- Total technique-paper references extracted: [X]
- References with explicit citations: [X]
- References requiring author-matching: [X]
- References excluded (no academic anchor): [X]

Stage 3: Paper Validation
- Candidate papers identified: [X]
- Excluded: paper does not introduce technique: [X]
- Excluded: paper not publicly accessible: [X]
- Excluded: duplicate (same paper, multiple artifacts): [X]

Stage 4: Final Sample
- Unique papers included for coding: [N]
- Papers with 1 adoption event: [X]
- Papers with 2+ adoption events: [X]
- Papers with regulatory adoption: [X]

Figure 1: PRISMA-style sample construction flow

Table 4: Papers extracted by artifact source

Category	Source	Techniques	Unique Papers
Tools	IBM ART	X	Y
	CleverHans	X	Y
	Foolbox	X	Y
	TextAttack	X	Y
	PyRIT	X	Y
Benchmarks	RobustBench	X	Y
	AutoAttack	X	Y
	HarmBench	X	Y
Regulatory	MITRE ATLAS	X	Y
	NIST AI RMF	X	Y
	OWASP Top 10	X	Y
Vendor	Cloud (AWS/Azure/GCP)	X	Y
	AI Labs (OpenAI/Anthropic)	X	Y
Total (before deduplication)		X	Y
Final sample (after deduplication)		—	**N**

]

6.2 RQ1: Adoption Lag Distributions

[PLACEHOLDER: [After coding, populate Table **??** with computed statistics]]

Table 5: RQ1: Summary of adoption lag distributions

Stratification	n	Median	IQR	Min	Max
Overall					
All papers	[N]	[X]	[Y–Z]	[X]	[Y]
By Artifact Type					
Tool integration	[n]	[X]	[Y–Z]		
Benchmark inclusion	[n]	[X]	[Y–Z]		
Regulatory citation	[n]	[X]	[Y–Z]		
Vendor acknowledgment	[n]	[X]	[Y–Z]		
By Publication Era					
2014–2017	[n]	[X]	[Y–Z]		
2018–2021	[n]	[X]	[Y–Z]		
2022–2025	[n]	[X]	[Y–Z]		

Key finding: Adoption lag has decreased by [X]% from 2014–2017 (median [X] months) to 2022–2025 (median [Y] months).]

6.3 RQ2: Domain Variation

Statistical comparison: Pairwise Mann-Whitney U tests with Bonferroni correction ($\alpha = 0.05/10 = 0.005$ for 5 domains).

Key finding: LLM security papers show significantly shorter adoption lags than computer vision papers (U = X.XX, $p < 0.001$), with median lag of X months vs. Y months. This represents an X-fold acceleration.

Table **??** shows the first adoption artifact type by domain, revealing that LLM techniques predominantly enter via vendor acknowledgment, while vision techniques enter via tools.]

Table 6: First adoption artifact by domain

[PLACEHOLDER:

Domain	Tool	Benchmark	Regulatory	Vendor
Vision	X%	Y%	Z%	W%
NLP (pre-LLM)	X%	Y%	Z%	W%
Malware	X%	Y%	Z%	W%
LLM	X%	Y%	Z%	W%

]

6.4 RQ3: Predictors of Adoption Speed

Table 7: Cox proportional hazards model: Predictors of time to first adoption

Covariate	HR	95% CI	p-value
Publication year (per year)	X.XX	[X.XX, X.XX]	<0.001
Code released (vs. not)	X.XX	[X.XX, X.XX]	X.XXX
Black-box (vs. white-box)	X.XX	[X.XX, X.XX]	X.XXX
Gray-box (vs. white-box)	X.XX	[X.XX, X.XX]	X.XXX
LLM domain (vs. vision)	X.XX	[X.XX, X.XX]	X.XXX
Security venue (vs. ML)	X.XX	[X.XX, X.XX]	X.XXX
Attack (vs. defense)	X.XX	[X.XX, X.XX]	X.XXX

Interpretation: Hazard ratio (HR) >1 indicates faster adoption. Each additional publication year is associated with X% faster adoption (HR = X.XX, 95% CI [X.XX, X.XX]), consistent with field maturation and improved infrastructure. Code release is associated with X% faster adoption (HR = X.XX, p = X.XXX). Black-box techniques show X% faster adoption than white-box techniques (HR = X.XX, p = X.XXX), supporting the hypothesis that realistic threat models accelerate industry uptake.

Model concordance: C-index = X.XX, indicating [good/moderate/poor] predictive discrimination.]

7 Factors Accelerating Adoption

Our analysis identifies four mechanisms accelerating research-to-practice transfer post-2020.

7.1 Standardized Evaluation

AutoAttack (**?**) and RobustBench (**?**) transformed evaluation practices by providing parameter-free, reproducible assessment. Before AutoAttack, reported robust accuracies were often inflated due to weak evaluation; AutoAttack reduced claims by >10% for 13 models. This standardization enabled practitioners to compare defenses on equal footing.

For LLMs, HarmBench (**?**) provides comparable standardization, evaluating 18 red-teaming methods across 33 models. MLCommons AILuminate (December 2024) extends this with 24,000+ prompts across 12 hazard categories, providing the first industry-standard safety benchmark.

7.2 Regulatory Mandates

The EU AI Act (**?**) Article 15 requires high-risk AI systems to achieve "appropriate level of accuracy, robustness and cybersecurity," with explicit requirements for "resilience regarding attempts by unauthorised third parties to alter their use." This creates compliance pressure driving adoption of adversarial testing.

NIST AI RMF (**?**) provides voluntary but widely-adopted guidance, with the Generative AI Profile (**?**) specifying 200+ actions including adversarial evaluation. Executive Order 14110 (October 2023) directed NIST to develop these guidelines, accelerating timeline.

7.3 Market Investment

The AI security market matured rapidly through major acquisitions:

- Cisco acquired Robust Intelligence for approximately $400M (August 2024)
- Palo Alto Networks acquired Protect AI for $500M+ (April 2025)
- F5 acquired CalypsoAI for $180M (September 2025)

Total acquisition value exceeds $1.1 billion in 2024–2025 alone. HiddenLayer raised $50M Series A (September 2023), the largest for an AI security startup that year. This capital enables productization of research techniques.

7.4 Direct User Adversarial Interaction

LLM security exhibits uniquely compressed adoption cycles because users directly interact with models and discover vulnerabilities. The Bing Chat "Sydney" incident saw system prompt extraction within 24 hours of launch. Jailbreaking communities share techniques in real-time. This creates pressure for rapid defense deployment absent in traditional ML where adversaries are more distant from model interfaces.

8 Discussion

8.1 Summary of Findings

[PLACEHOLDER: [Synthesize after completing analysis. Expected structure:]]

Our analysis of [N] papers spanning 2014–2025 reveals three principal findings:

1. **Adoption lag compression:** Overall adoption lag has decreased from [X–Y] years (2014–2017 papers) to [X–Y] years (2022–2025 papers), representing a [X]% reduction. This acceleration coincides with the emergence of standardized evaluation tools (AutoAttack, RobustBench) and regulatory mandates (EU AI Act, NIST AI RMF).
2. **LLM paradigm shift:** LLM security papers exhibit adoption lags of [X–Y] months—significantly shorter than computer vision ([X–Y] months, $p <$ [X.XXX]). This compression reflects direct user interaction with models, rapid vulnerability disclosure cycles, and intense commercial pressure.
3. **Predictors of adoption:** Cox regression identifies three significant predictors of faster adoption: (a) code availability at publication (HR = [X.XX], $p <$ [X.XXX]), (b) black-box threat model (HR = [X.XX], $p <$ [X.XXX]), and (c) publication in security venues vs. ML venues (HR = [X.XX], p = [X.XXX]).

8.2 Why Some Research Is Never Adopted

While our methodology focuses on adopted techniques, the artifact extraction process reveals structural barriers to adoption. Of X total techniques documented in academic papers but absent from our sample, we observe three patterns:

1. **Naming deficit:** Techniques without memorable names (e.g., "Method A," "Our approach") are difficult to trace and reference in practitioner contexts. All adopted techniques in our sample have distinctive names (FGSM, C&W, PGD, GCG).
2. **Tooling affordances:** Techniques requiring custom architectures, specialized hardware, or non-standard data formats face implementation barriers. Adopted techniques tend to be architecture-agnostic or provide reference implementations.
3. **Threat model compatibility:** White-box attacks requiring gradient access face adoption barriers in black-box deployment scenarios. Of X white-box techniques in our sample, Y% were adapted to gray-box or black-box variants before tool integration.

These observations suggest that adoption requires not only technical merit but also *adoptability*—the extent to which a technique can be named, implemented, and applied in practitioner workflows.

8.3 Implications for Researchers

Our findings suggest research design choices affect adoption probability:

- **Threat model realism:** Black-box and query-efficient methods show faster adoption than white-box approaches requiring gradient access
- **Code availability:** Code release correlates with tool integration (necessary but not sufficient condition)
- **Naming conventions:** Memorable technique names facilitate traceability and practitioner adoption
- **Standardized evaluation:** Benchmark inclusion accelerates uptake by enabling direct comparison
- **Real-world validation:** Testing on deployed systems, though rare, dramatically increases practitioner interest

Researchers seeking practical impact should consider threat models reflecting actual deployment constraints rather than worst-case theoretical assumptions.

8.4 Implications for Practitioners

The gap between research availability and production readiness suggests practitioners should:

- Monitor standardized benchmarks (RobustBench, HarmBench) for defense comparisons
- Leverage established toolkits (IBM ART, PyRIT) rather than implementing from papers
- Prioritize defenses validated under realistic threat models
- Anticipate regulatory requirements (EU AI Act compliance August 2026)

8.5 Limitations

Our analysis has several limitations. First, adoption evidence may be incomplete—commercial deployments often go undocumented. Second, our landmark paper selection, while systematic, involves judgment calls about impact thresholds. Third, we focus on English-language publications and Western regulatory frameworks. Fourth, the LLM era (2022–2025) provides limited longitudinal data for lag estimation.

9 Conclusion

This systematic review provides the first quantitative measurement of research-to-industry adoption lag in adversarial machine learning. Analyzing approximately 120 landmark papers from 2014–2025, we document adoption timelines across tool integration, commercial deployment, regulatory citation, and production use.

Our findings reveal domain-dependent patterns: traditional computer vision and malware detection exhibit 4–6 year lags; autonomous systems face 6–8 year timelines constrained by safety certification; LLM

security shows compressed 1–2 year cycles driven by direct user interaction and competitive pressure. Overall, adoption has accelerated substantially post-2020, driven by standardized evaluation (AutoAttack, RobustBench, HarmBench), regulatory mandates (EU AI Act, NIST AI RMF), and market investment ($1B+ in acquisitions).

The $1.1 billion in AI security acquisitions during 2024–2025 signals enterprise recognition that adversarial ML has transitioned from research curiosity to business requirement. Yet the gap between 5% attack experience and 86% security concern (**?**) indicates the market is positioning for threats that remain largely prospective. Bridging research and practice requires continued investment in realistic threat models, standardized evaluation, and accessible tooling.

Our coding framework and adoption evidence database are available at [PLACEHOLDER: [repository URL]] to support future research-practice alignment studies.

A Supplementary Materials

A.1 Data Dictionary

Table **??** provides the complete data dictionary for the released dataset.

Table 8: Data dictionary for released dataset

Column	Type	Description
Paper Identification		
paper_id	String	Unique identifier (DOI or arXiv ID)
title	String	Paper title
authors	String	Author list (semicolon-separated)
venue	String	Publication venue
pub_date	Date	Publication date (YYYY-MM-DD)
technique_name	String	Primary technique name
Coding Variables (see Table ??)		
G1–G7	Categorical	Research characteristics
T1–T4	Categorical	Threat model assumptions
Q1–Q3	Categorical	Practical evaluation indicators
Adoption Events		
tool_adoption_date	Date	First tool implementation (if any)
tool_name	String	Tool(s) implementing technique
benchmark_adoption_date	Date	Benchmark inclusion date (if any)
regulatory_adoption_date	Date	Regulatory citation date (if any)
vendor_adoption_date	Date	Vendor acknowledgment date (if any)
Computed Fields		
first_adoption_lag_months	Integer	Minimum lag across all events
median_adoption_lag_months	Float	Median lag across all events
regulatory_lag_months	Integer	Lag to regulatory citation (if any)
adoption_count	Integer	Number of distinct adoption events

A.2 Artifact Extraction URLs

- **IBM ART:** https://github.com/Trusted-AI/adversarial-robustness-toolbox
- **CleverHans:** https://github.com/cleverhans-lab/cleverhans
- **Foolbox:** https://github.com/bethgelab/foolbox
- **TextAttack:** https://github.com/QData/TextAttack
- **PyRIT:** https://github.com/Azure/PyRIT
- **RobustBench:** https://robustbench.github.io/
- **HarmBench:** https://github.com/centerforaisafety/HarmBench
- **MITRE ATLAS:** https://atlas.mitre.org/techniques
- **OWASP LLM Top 10:** https://owasp.org/www-project-top-10-for-large-language-model-app

A.3 Coding Examples

Example 1: FGSM (?)

- G1: Attack | G2: Evasion | G3: Vision | G4: ML | G5: Yes | G6: At-publication | G7: 2015

- T1: White-box | T2: Yes | T3: Zero (single forward-backward pass) | T4: L_∞
- Q1: No | Q2: Qualitative | Q3: No
- First adoption: CleverHans (October 2016) $\rightarrow$ 22 months lag

Example 2: GCG Attack (?)

- G1: Attack | G2: Evasion | G3: LLM | G4: ML | G5: Yes | G6: At-publication | G7: 2023
- T1: White-box | T2: Yes | T3: Unlimited | T4: Semantic
- Q1: Yes (tested on ChatGPT, Bard, Claude APIs) | Q2: Yes | Q3: Yes
- First adoption: HarmBench (February 2024) $\rightarrow$ 7 months lag

Example 3: Prompt Injection (?)

- G1: Attack | G2: Evasion | G3: LLM | G4: ML | G5: No | G6: Never | G7: 2022
- T1: Black-box | T2: No | T3: $<$100 | T4: Semantic
- Q1: Partial | Q2: No | Q3: No
- First adoption: OWASP LLM Top 10 (August 2023) $\rightarrow$ 9 months lag