

SECURITY

Prompt injection and the tool-using clinical assistant: a threat model

ABSTRACT

A language-model assistant in an ambulatory practice reads text written by outsiders, including faxes, referral letters, portal messages and web pages, and increasingly calls tools that write to the record or send messages. This review treats that arrangement as a confused-deputy problem and appraises the evidence. Models tested on medical tasks followed injected instructions, mostly at high rates. Defenses that detect or discourage injected text lowered attack rates on fixed test sets and were bypassed by adaptive attackers, and model developers describe the problem as unsolved. The controls that hold limit what a compromised model can do: tools fixed by the trusted request, human approval of every state-changing action, and output that cannot carry data out.

Type Threat model and literature review **References** 29 **Reading time** 15 min

Last reviewed September 2026

1 An assistant that reads what strangers write

A clinical assistant built on a language model is useful to the extent that it reads the record and acts on it. It summarizes a faxed referral packet, drafts a reply to a portal message, reconciles an outside medication list and, in configurations now being built and benchmarked, calls tools that query and write to the record through standard interfaces.¹ Most of what it reads was written by someone other than the clinician using it, some of it by people with no relationship to the practice at all.

Prompt injection is what happens when text inside that input is treated as an instruction. In direct injection the person typing into the assistant supplies the instruction. In indirect injection, the case that matters here, the instruction sits in data the application retrieves, and the attacker never touches the interface.² This paper asks what an attacker who can place text in front of the model can make it do, what the evidence shows, and which defenses survive testing.

The short answer has three parts. In every medical study reviewed here, the models tested could be induced to follow injected instructions, mostly at high rates.^{3,4,5} Defenses that detect or discourage injected text lower that rate on fixed test sets and fail against attackers who adapt to them,⁶ and the

companies that build the models describe the problem, in their own publications, as unsolved.^{7,8} What holds is architectural: limiting what a compromised model can do, so that a successful injection produces at worst a bad draft in front of a clinician rather than an action already taken.

2 The confused deputy, restated for language models

In 1988 Norm Hardy described a compiler that had permission to write files in its own directory, where it kept statistics.⁹ A user who learned the name of the billing file in that directory supplied it as the compiler's debugging-output file, and the compiler, acting on the user's request with its own permission, overwrote the billing records. Hardy's diagnosis was that the program served two masters, carried authority from each, and had no way to keep them apart.

A language model is that compiler with a larger vocabulary. Its input is a single stream of text in which the system's instructions, the clinician's request and the contents of a faxed letter are concatenated, and the model cannot reliably tell which part came from which source.¹⁰ The UK National Cyber Security Centre, in a December 2025 post by its CTO for Architecture, set this against SQL injection: a database distinguishes the instructions it executes from the data it stores, while inside a language model there is only the next token. The post concluded that "it's very possible that prompt injection attacks may never be totally mitigated in the way that SQL injection attacks can be," and proposed treating the model as an "inherently confusable deputy".¹¹ A classical confused deputy can be fixed. One that cannot be unconfused has to be given less authority.

Two taxonomies

Greshake and colleagues, who demonstrated indirect injection against a production search chatbot and applications built on GPT-4, sorted its consequences into information gathering, fraud, intrusion, malware, manipulated content and loss of availability.² NIST's adversarial machine learning taxonomy, in its 2025 edition, gives indirect prompt injection its own section and sorts it by the attacker's objective: availability, integrity and privacy compromise.¹² OWASP's list of risks for language-model applications, whose 2025 edition is still current, ranks prompt injection first and states that it is unclear whether fool-proof methods of prevention exist.¹³ These are voluntary frameworks and technical reports, not regulation.

3 Where untrusted text enters an ambulatory record

The attack surface of a clinical assistant is the set of channels through which text written by someone other than the clinician reaches the model. In an ambulatory practice those channels are numerous, and most are inbound by design.

TABLE 1

Channels through which untrusted text reaches a clinical assistant, with the closest published demonstration. None of these studies was conducted inside a clinical record system.

Channel	Who can write to it	Closest published evidence
Fax and scanned documents	Anyone with the number; text in an image becomes model input	Injected instructions, including sub-visual text inside medical images, raised lesion miss rates in all four models tested ³
Referral letters, outside records, uploaded PDFs	Outside clinicians, patients, anyone upstream of the sender	White and microscopic text hidden in 18 manuscripts to steer AI-assisted peer review ¹⁴
Patient portal messages	Any account holder	Injected instructions produced unsafe advice in 94.4% of injected simulated patient dialogues ⁴
Web pages fetched at run time	Site owners and contributors	15.3K validated injections across 11.7K web pages, about 70% in non-rendered HTML ¹⁵
Email from outside parties	Anyone	One email exfiltrated data from a production enterprise assistant with no user click ¹⁶
The record itself, once injected text is filed	Whoever wrote the text that was filed	An injection stored in an assistant's memory re-poisoned later sessions ²

Three features matter more than any single row. First, the channels are open on purpose. A practice publishes its fax number and gives every patient a portal account, so placing text in front of the assistant requires no intrusion.

Second, the instruction need not be visible to the person who reviews the document. OWASP notes that injected content need not be human-readable, provided the model parses it.¹³ Clusmann and colleagues reported that sub-visual prompts embedded in medical images were non-obvious to human observers.³ Lin documented instructions aimed at AI-assisted peer review, such as a command to give only a positive review, hidden in white text and microscopic fonts in 18 arXiv manuscripts in July 2025.¹⁴ Khodayari and colleagues, scanning 1.2 billion URLs, found about 70% of web injections in headers, comments and metadata that a browser never renders, though compliance in their controlled tests was at most 8%.¹⁵ A clinician looking at a fax image and a model reading its text layer are not reading the same document.

Third, the record is itself a channel. Text that is summarized or filed becomes context for later queries. Greshake's group demonstrated an injection that wrote itself into an assistant's memory and re-poisoned a later session that read it back,² and OWASP's list of risks for agentic applications, published in December 2025, names memory and context poisoning separately.¹⁷ An ambient transcript is also untrusted, since anyone in the room contributes to it, but none of the studies reviewed here tested spoken injection.

4 What an injected assistant can do

NIST's objectives map onto a clinical assistant with little translation.¹² Availability attacks, which make the assistant refuse or return nothing useful, matter least where a manual workflow exists. Integrity and confidentiality attacks matter more, and tools add a third category.

Integrity: the wrong content, plausibly written

The assistant produces a summary that omits a finding, a reconciliation that drops an allergy, or a draft reply that recommends something contraindicated. This is the class the medical studies test, and its only barrier is a reviewer who sees a fluent draft and has no signal that it was manipulated.

Confidentiality: data carried out in the output

The recurring mechanism is rendering: the model emits a link or image whose address encodes the data, and the client fetches it. Greshake demonstrated markdown links that hide a suspicious address behind innocent text,² and one of OWASP's example scenarios is a summarized web page that makes the model insert an image linking to an attacker's URL.¹³ A documented production case is EchoLeak, CVE-2025-32711, scored 9.3 by Microsoft. A crafted email, retrieved when a user later queried an enterprise assistant, caused it to place data in a reference-style markdown image that the client fetched automatically, through an allowlisted proxy that satisfied the content security policy. The email evaded Microsoft's injection classifier by reading as an ordinary request to its human recipient. Microsoft fixed it server-side in May 2025 and reported no evidence of exploitation in the wild.¹⁶

For a HIPAA covered entity the consequence is familiar. Protected health information carried out this way is an acquisition or disclosure not permitted by the Privacy Rule, presumed under 45 C.F.R. § 164.402 to be a breach unless a four-factor risk assessment demonstrates a low probability of compromise.¹⁸ The EchoLeak attacker needed only the ability to send an email.

Action: the instruction executed

When the assistant holds tools, the injected instruction can be carried out rather than merely written. InjecAgent's 1,054 test cases divide attacker goals into direct harm to users and exfiltration of private data.¹⁹ The developers' own examples are instructive because they are mundane. OpenAI described a malicious email that caused a browsing agent, asked to draft an out-of-office reply, to send a resignation letter to the user's chief executive instead.⁷ Anthropic reported a test email claiming messages had to be deleted for security reasons, which its browsing agent obeyed without confirmation before mitigations were added.⁸ The clinical equivalents are a message sent to the wrong party, an order queued, a result released, an amendment filed: the actions a tool-using clinical assistant is built to take.

5 The clinical evidence: small, simulated, and uniformly positive

94.4% of injected simulated patient dialogues yielded unsafe advice at the decision point	22% → 89% GPT-4o lesion miss rate, unaltered versus injected prompts	12 recent defenses bypassed by adaptive attacks, most at over 90% success
---	--	---

Three peer-reviewed studies located for this review test injection against models performing medical tasks. Each found the models it tested vulnerable.

Clusmann and colleagues ran 594 attacks against four vision-language models, Claude 3 Opus, Claude 3.5 Sonnet, GPT-4o and Reka Core, on 18 oncology cases in six imaging modalities, placing the instruction in the text prompt or inside the image.³ Success was scored as a missed lesion. GPT-4o’s lesion miss rate rose from 22% with unaltered prompts to 89% with injection, and the rise was significant for every model (p from 0.02 to < 0.001). Two mitigations were tried, alone and in combination: a sentence telling the model to adhere to the highest ethical standards, and a second model instance acting as supervisor. None protected three of the four models; the ethics sentence reduced Claude 3.5 Sonnet’s measured vulnerability from 64.8% to 27.8%. The models were accessed between June and September 2024.

Lee and colleagues, in a study *JAMA Network Open* classifies as quality improvement, ran 216 simulated patient dialogues, 108 with injection and 108 controls, across 12 scenarios stratified by harm and three lightweight commercial models, inserting the injection programmatically.⁴ It succeeded in 94.4% of evaluations (102 of 108) and persisted in 69.4% of follow-ups. Extremely high-harm scenarios, including FDA Category X drugs in pregnancy such as thalidomide, succeeded in 91.7% (33 of 36). A proof-of-concept delivering the injection through client-side code succeeded against three flagship models in 5 of 5, 5 of 5 and 4 of 5 dialogues.

Yang and colleagues modeled a third-party application between user and model that appends a malicious instruction to the system prompt, unseen by the user.⁵ On MIMIC-III patient data, the attack moved GPT-4o’s rate of recommending COVID-19 vaccination from 88.06% to 6.47% and its rate of recommending a dangerous drug combination from 1.00% to 61.19%.

WHAT THE MEDICAL STUDIES DO NOT ESTABLISH

They show that models of 2024 and 2025 followed injected instructions on medical tasks at high rates and that the resulting output was clinically plausible. They give no rate for any deployed product. The investigators placed the instruction themselves, so the studies measure compliance given delivery, not the likelihood of delivery; the scenario sets are small; the models have since been replaced; and none tested an assistant with tools acting on a clinical record. Data poisoning, in which replacing 0.001% of training tokens with medical misinformation produced more harmful models that still matched clean ones on standard medical benchmarks, is a separate training-time attack outside this review.²⁰

The record-connected case is untested in the peer-reviewed literature located for this review. MedAgentBench, 300 physician-written tasks in a FHIR environment, measures capability, not robustness.¹ The one healthcare-specific injection benchmark found is a June 2026 preprint by the developers of the firewall it evaluates, whose agent experiment ran 40 attack episodes on one small open model.²¹ Its useful contribution is an observation: requests that harm a clinical system, such as reading another patient's record or exporting in bulk, look legitimate and carry no attack signal. On its 2,000-prompt healthcare benchmark, Meta's PromptGuard-2 classifier had a recall of 0.40.

6 Filters lower a rate; they do not bound a consequence

The larger general agent literature points the same way. InjecAgent found a ReAct-prompted GPT-4 agent vulnerable in 24% of cases, nearly doubling when the attacker's instruction was reinforced with a hacking prompt.¹⁹ In AgentDojo's 97 tasks and 629 security test cases, GPT-4o completed 69.0% of tasks without attack and 50.1% under attack, and carried out the attacker's goal in 47.7% of cases under a generic "important message" injection.²²

What the defenses measured

In AgentDojo, a classifier that detects injected text had too many false positives and significantly degraded utility. A tool filter, which the authors called particularly effective, restricted the agent to the tools the user's task needs before it saw any untrusted data, lowering attack success to 7.5%, because in many test cases the user's task needed only read access while the attacker's needed write access. It fails when the needed tools cannot be planned in advance or when they suffice for the attack.²² The defense the authors singled out restricted capability rather than inspecting text.

Prompt-level defenses report strong numbers against fixed attacks. Spotlighting, a family of techniques from Microsoft researchers that mark untrusted input by delimiting, datamarking or encoding it, cut attack success with GPT-family models from above 50% to below 2%.¹⁰ Against attackers who adapt, such numbers do not survive. Zhan and colleagues bypassed all eight defenses for tool-using agents they evaluated, with attack success above 50%.²³ Nasr and colleagues, whose authors include researchers at Anthropic, Google DeepMind and OpenAI, attacked 12 recent defenses based on prompting (spotlighting among them), training, detection models and secret knowledge. Attack success exceeded 90% for most, though the majority had originally reported near-zero attack success; three of four detection models were bypassed at above 90% and the fourth at 71%; and human red-teamers in a competition of more than 500 participants defeated every challenge.⁶ An instruction to disregard injected text, or to behave ethically, is a defense of the same kind; Clusmann's ethics sentence partly protected one model of four.³

What the developers say

OpenAI wrote in December 2025 that "prompt injection, much like scams and social engineering on the web, is unlikely to ever be fully 'solved'."⁷ Anthropic reported in August 2025 an attack success rate for its browsing agent of 23.6% without mitigations and 11.2% with them in autonomous mode, across 123 test cases. In November 2025, comparing the browser extension it was then launching with its original configuration against an internal adaptive attacker given 100 attempts per environment, it reported that Claude Opus 4.5 was more robust than earlier models, wrote that a 1% attack success rate still represents meaningful risk, and described prompt injection as far from solved.⁸ Microsoft calls indirect injection an inherent risk of probabilistic language modeling and says its defenses do not rely on blocking every injection.²⁴ Meta calls it a fundamental, unsolved weakness in all language models.²⁵ These are statements about the companies' own products, not independent measurements; their direction is the notable part.

WHAT AN ATTACK SUCCESS RATE MEASURES

A reported rate belongs to one model version, one attack set and one attacker budget. Developers' reported rates have fallen with successive mitigations and models, and the adaptive-attack literature shows that rates measured against fixed attack sets overstate robustness. A practice's exposure grows with the number of attempts it receives, and the attacker chooses that number. No figure here transfers to a product not tested in its deployed configuration.

7 The controls that hold are limits on capability

Saltzer and Schroeder's 1975 statement of least privilege is that every program and every user should operate with the least set of privileges necessary to complete the job, and complete mediation requires every access to every object to be checked for authority.²⁶ The NCSC applies this directly: injection into a system with tools has the impact of giving an attacker direct access to those tools, so a model processing email from arbitrary outsiders should not hold privileged tools.¹¹ OWASP traces excessive agency to excessive functionality, permissions and autonomy.¹³

Meta's "Agents Rule of Two" puts the constraint in a form a buyer can apply. Within a session an agent should have no more than two of three properties: it processes untrustworthy inputs; it has access to sensitive systems or private data; it can change state or communicate externally. An agent needing all three should not operate autonomously and requires, at a minimum, human-in-the-loop approval or another reliable means of validation.²⁵ A clinical assistant has the first two by construction, because its inputs are faxes and portal messages and its context is the chart. By the rule, every action in the third category then needs human approval or another reliable means of validation.

The research literature is turning this into architecture. Fourteen authors from groups including Google, IBM, Microsoft and ETH Zurich set out design patterns, such as plan-then-execute and a privileged model paired with a quarantined one, on the principle that once an agent "has ingested untrusted input, it must be constrained so that it is impossible for that input to trigger any consequential actions".²⁷ They are candid about cost: in the strong form of their patient-facing diagnostic case study, the response cannot react to what the patient said. CaMeL derives control and data flow from the trusted request so that retrieved data cannot change the program, and checks policy when tools are called; it solved 77% of AgentDojo tasks with provable security, against 84% undefended.²⁸

The HIPAA Security Rule already frames tool access this way. Its access control standard at 45 C.F.R. § 164.312(a)(1) limits access to electronic protected health information to "persons or software programs" granted access rights, and § 164.312(b) requires mechanisms that record and examine activity.²⁹ A tool-calling assistant is a software program holding access rights, and its grants belong in the same access design as a user's.

Properties that follow

1. **Tools are fixed by the trusted request, not by what the model reads.** The callable tools are set before untrusted content enters the context, as AgentDojo's tool filter and CaMeL both do.^{22,28}
Summarizing an outside record needs read access to that record and nothing else.
2. **Scope is the patient and task in hand.** The assistant acts with the requesting clinician's permissions and no more,¹³ and cannot reach another patient's chart or a bulk export, the requests a detector is least able to flag.²¹

3. **Every state-changing or outbound action waits for a named person.** Sending, ordering, releasing and amending are proposals until approved, as where the user must approve and send generated email text themselves.²⁴ An injected assistant can also write the justification the approver reads, which OWASP calls human-agent trust exploitation,¹⁷ so the approval shows the exact action and the source documents in context.
4. **Untrusted content is marked, for people and for policy.** Provenance labels are not a model-level defense; spotlighting was among the defenses bypassed adaptively.^{6,10}
5. **Output cannot carry data out.** No automatically fetched remote images, no links to unapproved destinations, an allowlist for network egress. Microsoft reports deterministically blocking the markdown-image exfiltration technique itself rather than relying on catching the injection,²⁴ and EchoLeak shows what one allowlisted proxy can undo.¹⁶
6. **Filed text does not become standing instruction.** Content derived from outside documents keeps its source in the record and re-enters later contexts as data.^{2,17}
7. **Every proposed action is logged with the inputs that were in context,** so that a successful injection can be traced to the document that carried it.²⁹

8 What a buyer can verify, and what would settle the question

Most of these properties can be checked without access to the model. A buyer can ask which tools the assistant can call and with whose permissions; whether anything arriving by fax, portal, email or web can cause a tool call, message or order without a clinician's approval; whether output can render remote images or links; whether testing used adaptive attackers or a fixed set; and what the audit log records. A vendor that answers with a detection rate has answered a different question.

Four statements are supported. Models on medical tasks followed injected instructions at high rates.^{3,4,5} Instructions can be hidden from the human reader of the same document.^{3,14,15} Defenses that detect or discourage injected text fail against adaptive attackers, and the model developers do not claim otherwise.^{6,7,8,23} The defenses that bound what an injection can do restrict capability, and they narrow what the agent can complete.^{22,28}

Three kinds of evidence would settle what remains open: adaptive red-teaming of deployed clinical assistants in their real configurations, with the attacker's budget reported; benchmarks that give a tool-using agent a realistic record, realistic inbound documents and injected content together; and reports of injection incidents from clinical deployments, of which the literature reviewed here contains none. Until then the defensible position is the architectural one. A clinical assistant that reads what strangers write should be assumed steerable by them, and built so that being steered changes a draft, not the record.

REFERENCES

Entries 3, 4 and 5 are the peer-reviewed tests of injection on medical tasks and carry section 5; entries 6, 19, 22 and 23 are the agent benchmarks and adaptive-attack evaluations that carry section 6. Entries 10, 15, 21, 27 and 28 are preprints and are flagged in the list; entry 21 is authored by the developers of the tool it evaluates, and entry 4 is described by its journal as a quality improvement study. Entries 7, 8, 24 and 25 are statements by model developers about their own products, not independent measurements; entry 11 is a national cyber-security agency's published analysis, and entries 12, 13 and 17 are voluntary frameworks or technical reports; entries 18 and 29 are regulation, cited for what they require rather than as evidence. Pages could not be confirmed for entries 1 and 6, and the task counts quoted from entry 1 are taken from its preprint.

1. Jiang Y, Black KC, Geng G, et al. *MedAgentBench: A Virtual EHR Environment to Benchmark Medical LLM Agents*. NEJM AI. 2025;2(9). [doi:10.1056/Aldbp2500144](https://doi.org/10.1056/Aldbp2500144). Task counts taken from the preprint, [arXiv:2501.14654](https://arxiv.org/abs/2501.14654), and the project page; pages of the journal version not confirmed.

BENCHMARK

2. Greshake K, Abdelnabi S, Mishra S, et al. *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 '23). 2023:79–90. [doi:10.1145/3605764.3623985](https://doi.org/10.1145/3605764.3623985)

CASE STUDY

3. Clusmann J, Ferber D, Wiest IC, et al. *Prompt injection attacks on vision language models in oncology*. Nature Communications. 2025;16:1239. [doi:10.1038/s41467-024-55631-x](https://doi.org/10.1038/s41467-024-55631-x)

BENCHMARK

4. Lee RW, Jun TJ, Lee JM, et al. *Vulnerability of Large Language Models to Prompt Injection When Providing Medical Advice*. JAMA Network Open. 2025;8(12):e2549963. [doi:10.1001/jamanetworkopen.2025.49963](https://doi.org/10.1001/jamanetworkopen.2025.49963)

BENCHMARK

5. Yang Y, Jin Q, Huang F, et al. *Adversarial prompt and fine-tuning attacks threaten medical large language models*. Nature Communications. 2025;16:9011. [doi:10.1038/s41467-025-64062-1](https://doi.org/10.1038/s41467-025-64062-1)

BENCHMARK

6. Nasr M, Carlini N, Sitawarin C, et al. *The Attacker Moves Second: Stronger Adaptive Attacks Bypass Defenses Against LLM Jailbreaks and Prompt Injections*. 35th USENIX Security Symposium (USENIX Security '26), August 2026; pages not confirmed. usenix.org; preprint [arXiv:2510.09023](https://arxiv.org/abs/2510.09023)

BENCHMARK

7. OpenAI. *Continuously hardening ChatGPT Atlas against prompt injection attacks*. 22 Dec. 2025. openai.com

INDUSTRY STATEMENT

8. Anthropic. *Piloting Claude in Chrome*, 25 Aug. 2025, claude.com; and *Mitigating the risk of prompt injections in browser use*, 24 Nov. 2025, anthropic.com

INDUSTRY STATEMENT

9. Hardy N. *The Confused Deputy (or why capabilities might have been invented)*. ACM SIGOPS Operating Systems Review. 1988;22(4):36–38. [doi:10.1145/54289.871709](https://doi.org/10.1145/54289.871709) POSITION PAPER
10. Hines K, Lopez G, Hall M, et al. *Defending Against Indirect Prompt Injection Attacks With Spotlighting*. arXiv. 2024. [arXiv:2403.14720](https://arxiv.org/abs/2403.14720). Authors affiliated with Microsoft. PREPRINT
11. Chismon D; National Cyber Security Centre (UK). *Prompt injection is not SQL injection (it may be worse)*. NCSC blog, 8 Dec. 2025. ncsc.gov.uk POSITION PAPER
12. Vassilev A, Oprea A, Fordyce A, et al. *Adversarial Machine Learning: A Taxonomy and Terminology of Attacks and Mitigations*. NIST AI 100-2 E2025. National Institute of Standards and Technology; March 2025. [doi:10.6028/NIST.AI.100-2e2025](https://doi.org/10.6028/NIST.AI.100-2e2025) GOVERNMENT REPORT
13. OWASP Gen AI Security Project. *2025 Top 10 Risk & Mitigations for LLMs and Gen AI Apps*, entries *LLM01:2025 Prompt Injection* and *LLM06:2025 Excessive Agency*. genai.owasp.org/llmrisk/llm01; genai.owasp.org/llmrisk/llm06 STANDARD
14. Lin Z. *Hidden Prompts in Manuscripts Exploit AI-Assisted Peer Review*. Communications of the ACM. 2026;69(7):53–56. [doi:10.1145/3779116](https://doi.org/10.1145/3779116); preprint [arXiv:2507.06185](https://arxiv.org/abs/2507.06185) CASE STUDY
15. Khodayari S, Zhang X, Acharya B, et al. *Indirect Prompt Injection in the Wild: An Empirical Study of Prevalence, Techniques, and Objectives*. arXiv. 2026. [arXiv:2604.27202](https://arxiv.org/abs/2604.27202) PREPRINT
16. Reddy P, Gujral AS. *EchoLeak: The First Real-World Zero-Click Prompt Injection Exploit in a Production LLM System*. Proceedings of the AAAI Symposium Series. 2025;7(1):303–311. [doi:10.1609/aaais.v7i1.36899](https://doi.org/10.1609/aaais.v7i1.36899). The vulnerability is CVE-2025-32711 (CNA Microsoft; CVSS 3.1 base score 9.3; published 11 June 2025), cve.org CASE STUDY
17. OWASP Gen AI Security Project. *OWASP Top 10 for Agentic Applications 2026*, released 9 Dec. 2025; entries ASI06 Memory & Context Poisoning and ASI09 Human-Agent Trust Exploitation. genai.owasp.org STANDARD
18. U.S. Department of Health and Human Services. *Definitions (breach)*. 45 C.F.R. § 164.402 (eCFR, current as of Sept. 24, 2026). ecfr.gov REGULATION
19. Zhan Q, Liang Z, Ying Z, et al. *InjecAgent: Benchmarking Indirect Prompt Injections in Tool-Integrated Large Language Model Agents*. Findings of the Association for Computational Linguistics: ACL 2024. 2024:10471–10506. [doi:10.18653/v1/2024.findings-acl.624](https://doi.org/10.18653/v1/2024.findings-acl.624) BENCHMARK
20. Alber DA, Yang Z, Alyakin A, et al. *Medical large language models are vulnerable to data-poisoning attacks*. Nature Medicine. 2025;31(2):618–626. [doi:10.1038/s41591-024-03445-1](https://doi.org/10.1038/s41591-024-03445-1) BENCHMARK

21. Schwoebel J, Semeneć I, Rousseva J, et al. *Beyond Injection Detection: A Positive-Security Prompt Firewall that Closes the Scope and PHI Gap SOTA Classifiers Miss in Healthcare*. medRxiv. Posted 11 June 2026. medrxiv.org. Evaluates a tool built by its authors. PREPRINT
22. Debenedetti E, Zhang J, Balunović M, et al. *AgentDojo: A Dynamic Environment to Evaluate Prompt Injection Attacks and Defenses for LLM Agents*. Advances in Neural Information Processing Systems 37 (NeurIPS 2024), Datasets and Benchmarks Track. 2024:82895–82920. proceedings.neurips.cc; figures from [arXiv:2406.13352v3](https://arxiv.org/abs/2406.13352v3) BENCHMARK
23. Zhan Q, Fang R, Panchal HS, et al. *Adaptive Attacks Break Defenses Against Indirect Prompt Injection Attacks on LLM Agents*. Findings of the Association for Computational Linguistics: NAACL 2025. 2025:7116–7132. [doi:10.18653/v1/2025.findings-naacl.395](https://doi.org/10.18653/v1/2025.findings-naacl.395) BENCHMARK
24. Pavard A. *How Microsoft defends against indirect prompt injection attacks*. Microsoft Security Response Center blog, 29 July 2025. microsoft.com INDUSTRY STATEMENT
25. Meta. *Agents Rule of Two: A Practical Approach to AI Agent Security*. Meta AI blog, 31 Oct. 2025. ai.meta.com INDUSTRY STATEMENT
26. Saltzer JH, Schroeder MD. *The protection of information in computer systems*. Proceedings of the IEEE. 1975;63(9):1278–1308. [doi:10.1109/PROC.1975.9939](https://doi.org/10.1109/PROC.1975.9939) REVIEW
27. Beurer-Kellner L, Buesser B, Crețu A-M, et al. *Design Patterns for Securing LLM Agents against Prompt Injections*. arXiv. 2025. [arXiv:2506.08837](https://arxiv.org/abs/2506.08837) PREPRINT
28. Debenedetti E, Shumailov I, Fan T, et al. *Defeating Prompt Injections by Design*. arXiv. 2025. [arXiv:2503.18813](https://arxiv.org/abs/2503.18813) PREPRINT
29. U.S. Department of Health and Human Services. *Technical safeguards*. 45 C.F.R. § 164.312(a)(1), (b) (eCFR, current as of Sept. 24, 2026). ecfr.gov REGULATION

RELATED PAPERS

Approval gates

Where human decision authority has to sit before an assistant's action takes effect.

A write-path architecture

What may enter the record, in what state, and on whose authority.

PHI and the model

What HIPAA requires when protected health information reaches a model.