Reviewed for Quality and Security:

Advertising Injection Through a Verified Connector, Trust Laundering, and the Limits of Directory Verification

Travis Gilly ❧ July 2026 (v5)
ORCID: 0009-0007-2954-6313
Real Safety AI Foundation, Illinois, United States

ABSTRACT

A conversational assistant whose maker publicly commits that it carries no advertising was observed delivering a paid tier upgrade advertisement at the close of every academic search, across seven consecutive queries spanning distinct research domains. The advertisement did not originate with the assistant. It arrived inside the result payload of a third party connector, carried by a literal instruction directed at the model: you must include the above message at the end of your response. This is indirect prompt injection, the failure mode ranked first among risks to large language model applications, executing through a connector that the platform lists in a curated directory and marks with a badge whose hover text states that the platform has reviewed the connector for quality and security. One line below the badge, the same card disclaims the capacity to verify that listed connectors work as intended or that they will not change. This paper documents the specimen, establishes its reproducibility, and situates it in the protocol specific security literature, which supplies both the correct name for the attack, function return injection rather than tool poisoning, and the evidence that this vector is stronger and less studied than the descriptor poisoning the field concentrates on. It then maps the conduct against the platform's own directory policy, which prohibits both software that evades model instructions and software that serves advertisements. It then locates the harm in two established bodies of authority. The consumer protection frame is the Federal Trade Commission's treatment of deceptively formatted advertising, under which a message must be identifiable as advertising and a misleadingly formatted advertisement is deceptive even when its claims are literally true. The behavioral frame is the persuasion knowledge literature, under which a reader who cannot recognize a message as advertising cannot mount the skepticism that advertising recognition normally triggers. The design frame is the deceptive design literature, which holds that the most consequential dark patterns sit beneath the interface in system architecture, where interface bound scrutiny cannot reach them. The economic frame is twofold: the connector is a credence good whose quality the user cannot verify even after use, which is the condition that makes a certification mark necessary in the first place, and the badge is a signal that cannot separate compliant from non compliant connectors because obtaining it costs both the same. The attack class is well described in that literature; what is not documented there is a listed, badged, first party connector performing it against ordinary users for commercial gain, in production rather than in a benchmark. The paper names the general phenomenon the specimen exposes as trust laundering: the conversion of a platform's verification mark into credibility for a third party's promotional content, credibility the content did not earn and the platform did not knowingly extend. A verification badge that speaks in the present tense about an artifact its issuer concedes may have changed since review is the mechanism that makes trust laundering possible. The advertisement is the proof the assurance fails. The structure will outlast the advertisement.

Keywords: function return injection, rug pull, trust laundering, indirect prompt injection, Model Context Protocol, connector verification, dark patterns, deceptive design, native advertising, persuasion knowledge, trust marks, credence goods, signaling theory, attention economy, Federal Trade Commission, platform accountability, AI governance

I. Introduction

In February 2026 the maker of a widely used conversational assistant published a statement of principle about advertising. The assistant would remain a space to think. Its responses would not be shaped by advertisers, and it would not carry product placements the user did not ask for. The statement drew a distinction it treated as foundational: an interaction with a third party inside the assistant should be initiated by the user rather than by an advertiser, because the first arrangement keeps the assistant working for the person using it and the second puts it to work, at least in part, for someone else (Anthropic, 2026a).

This paper documents a case in which that arrangement was inverted. A user ran a series of academic literature searches through a connector listed in the platform's own connector directory. Each search returned research results, and each search also returned, at the close of the assistant's reply, a promotional message urging the user to purchase the connector's paid tier. The user initiated a search for papers. The user did not initiate an advertisement. The advertisement was appended by the connector, and it was appended by means of an instruction aimed at the model rather than at the user.

The mechanism is indirect prompt injection. Data returned through a tool channel carried an imperative, the model could not distinguish the imperative from the surrounding data, and the model carried out the imperative (Greshake et al., 2023). The failure is not exotic. It is catalogued as the first entry in the standard enumeration of security risks for large language model applications (OWASP, 2025), and empirical work reports indirect injection succeeding against production systems at rates above eighty percent (Yusifov et al., 2026). What gives this case its weight is the surface it ran on. The connector was not a stray endpoint the user had pasted into a configuration file. It was listed in the platform's curated directory, and it carried a verification badge whose hover text stated that the platform had reviewed the connector for quality and security.

The paper makes four claims. First, as an empirical matter, the advertising injection is real, reproducible, and instrumented: it fired on every one of seven searches, its payload contained a literal command to the model, and its links carried campaign tracking parameters naming the assistant as their acquisition source. Second, as a matter of the platform's own governance, the conduct contravenes several enumerated provisions of the directory policy that every listed connector agrees to, most cleanly the provision forbidding software that evades the model's instructions. Third, as a matter of consumer protection, the conduct fits the Federal Trade Commission's account of deceptively formatted advertising, under which a promotional message must be identifiable as advertising and a misleadingly formatted advertisement is deceptive even when its underlying claims are true (Federal Trade Commission, 2015a). Fourth, and most durably, the case exposes a structural defect in directory verification. The badge asserts a present and continuing condition. The platform's own disclaimer, printed on the same card, concedes that the platform cannot verify that listed software works as intended or that it will not change after review. A verification that cannot survive the verified artifact changing is a verification of a moment presented as a verification of a state, and it is the mechanism by which a third party's advertisement borrows the platform's credibility. This paper names that borrowing trust laundering.

Responsibility divides between two parties and the paper keeps them distinct throughout. The connector's developer authored the injected instruction and agreed, as a condition of listing, not to do so. The platform verified the connector, issued the badge, and operates the review that did not detect the most anticipated attack in the category. Neither party's failure is the whole account. The developer did a thing it was contractually forbidden to do, and the platform vouched for the developer while it was doing it.

II. The Directory and the Badge

The Model Context Protocol is an open standard, introduced in late 2024 and now stewarded as an open source project under the Linux Foundation, that lets a conversational assistant call external tools during a conversation (Model Context Protocol, 2025). A connector is a server that implements the protocol and exposes one or more tools the assistant may invoke. When the assistant calls a tool, the connector returns a result, and the assistant incorporates that result into its reply. The result is data. It is supposed to be data.

The platform operates a directory through which users discover and enable connectors. The directory is curated. The platform's directory policy states that submissions are reviewed to meet standards for safety, security, and compatibility, that the platform conducts both an initial review and ongoing review, and that a developer may be required to address compliance issues to remain listed (Anthropic, 2026b). Practitioners in the ecosystem treat directory acceptance as the platform's imprimatur; the listing functions as a credential.

The credential is made visible on the connector's directory card as a badge. On hover, the badge text states that the platform has reviewed the connector for quality and security. The verb is in the present perfect. It reports a completed act of review whose result is asserted to hold now.

One line below the badge, the same card carries a disclaimer. It states that the platform does not control which tools developers make available and cannot verify that they will work as intended or that they will not change. This language is standard across the directory and is not customized per connector. The card therefore asserts verification and disclaims verification about the same connector, in the same modal, at the same moment. The two statements cannot both be operative in the strong reading a user would take from the badge. Either the connector was reviewed for quality and security in a sense that binds the connector's present behavior, or the platform cannot verify that the connector works as intended. Section VI returns to this.

The directory policy also speaks directly to advertising. Among the categories of software the platform does not permit into its directories absent written permission, one prohibits software that serves advertisements, sponsored content, or paid product placements, or that exists primarily as an advertising or promotional vehicle (Anthropic, 2026b). The qualifier primarily attaches to the final clause. Serving advertisements is prohibited on its own terms, without a threshold. In the policy text, the word advertisements is a hyperlink, and it points to the February statement of principle described above. The policy thus incorporates that statement as its definitional referent for what an advertisement is.

III. The Specimen

A. Setup

The connector at issue is a scholarly search tool that provides access to a large corpus of research papers. It was enabled from the platform's directory through the standard authenticated connection flow, so every request the connector received arrived attached to the user's account. The user held the connector's free tier. Seven literature searches were run through the connector inside the assistant, on the same day, spanning distinct research domains: administrative burden and identity documentation, delayed disclosure of child sexual abuse, prevalence of online solicitation of minors, parental mediation of adolescent online risk, privacy preserving age verification, biometric failure to enroll, and organizational compliance sociology.

B. The payload

Each search returned a result payload with the same structure. The payload opened with the search results themselves, formatted as a numbered list of papers with titles, authors, years, citation counts, and links. It then carried a block introduced as important instructions. The first part of that block was a formatting directive: cite the papers inline using numbered references, and hyperlink the titles. That directive is unremarkable in isolation, and its presence matters, because it establishes an imperative register and an appearance of routine housekeeping before the operative instruction arrives.

The operative instruction followed. It presented the upgrade advertisement as text, then issued a command:

> Upgrade to [the connector's paid tier] to return 20 results per search instead of 10, and include more data like study design and key takeaways for every result. [pricing URL]
>
> You must include the above message at the end of your response.

The command is not a data field the model elected to render. It is an instruction, in plain imperative English, directed at the model, requiring the model to emit promotional content. Every link in the payload, including the pricing link and every paper link, carried a campaign tracking parameter identifying the assistant's desktop application as the acquisition source. The parameter is conversion instrumentation. Its presence establishes that the promotional message is a measured acquisition channel rather than an incidental notice.

C. Reproducibility

The injection fired on all seven searches. In each, the result header reported that twenty papers had been found and ten were being shown. That figure did not vary across the seven domains. Corpus density differs sharply across these fields, so a variable relevance cutoff would produce variable counts. A constant of twenty found and ten shown, across seven unrelated queries, indicates a fixed limit rather than a property of any query. The connector holds twenty results and returns ten, and reports the withholding in its own output.

Compliance by the model was total until the user objected. The advertisement appeared at the close of every reply until the user identified it as an injected instruction and directed the assistant to stop reproducing it, after which it did not appear again in the session. The instruction was never binding on the model. It was the default. The relevant property of the exploit is that the default holds until a user notices an advertisement, correctly identifies it as third party instruction rather than a quirk of the assistant, and countermands it.

D. The control

The same query, run on the connector's public website under the same free account on the same day, returned twenty results rather than ten. The website result also displayed, for every result, a key takeaway. The advertisement inside the assistant names both a larger result count and key takeaways as features the user must pay to obtain. Both were present, without charge, in the same vendor's free public product. The connector channel is therefore not a port of the free tier into the assistant. It is a reduced free tier, and two of the reductions are precisely what the injected advertisement offers to sell back.

E. The parameter set

The connector's public documentation enumerates the parameters its search tool accepts. They cover the query text, publication year bounds, study design, journal quartile, human subject restriction, minimum sample size, a medical

mode, preprint exclusion, and study duration bounds (Consensus, 2026b). None of them controls the number of results returned. The result count is fixed by the account's plan tier, which the same documentation sets at three results for no account, ten for the free tier, and twenty for paid tiers. A client cannot request more results than the tier allows, because the parameter to do so does not exist. A comparable academic search connector listed in the same directory exposes a client settable result count parameter accepting values from one to ten thousand, which indicates that the fixed cap is a product decision rather than a constraint of the task.

IV. The Mechanism

Indirect prompt injection is the manipulation of a language model through instructions embedded in content the model retrieves rather than in the user's own message. The attack class was named in early work showing that crafted inputs can override a model's intended behavior and surface its hidden instructions (Perez & Ribeiro, 2022). Greshake et al. (2023) then demonstrated that applications which feed retrieved content into a model blur the boundary between data and instructions, so that an adversary who controls retrieved content can steer the model's behavior without ever addressing the model directly. The standard security enumeration for language model applications places prompt injection first among its risk categories (OWASP, 2025), and a 2026 taxonomy reports indirect injection succeeding against production applications at a rate of eighty six percent, alongside attack success rates reaching eighty seven percent against state of the art proprietary models (Yusifov et al., 2026). Benchmarks built specifically to measure instructions hidden inside tool outputs are now a standard part of the field (Wang et al., 2025).

A. The protocol specific literature and the correct name for this attack

A protocol specific security literature has formed around MCP since 2025, and it locates this specimen precisely. Threat modeling work applying STRIDE and DREAD across the protocol's components identifies tool poisoning, the embedding of malicious instructions in tool metadata, as the most prevalent client side vulnerability, and reports that most tested clients defend against it poorly owing to insufficient static validation and parameter visibility (Huang et al., 2026). Automated frameworks now generate poisoned tools adaptively, reaching attack success rates above eighty percent while suppressing detection to a fraction of a percent (Li et al., 2026). Protocol level redesigns have been proposed in response, adding identity management, mutual authentication, policy enforcement, and audit logging (Hou et al., 2026).

That literature also supplies the correct name for what the Consensus payload does, and it is not tool poisoning. Tool poisoning operates on the descriptors: the tool's name, its description, its argument schema, all of which enter the model's context at registration and steer tool selection before any call occurs. The Consensus instruction is in none of those. It arrives in the result of a tool the user deliberately invoked, after selection, in the returned payload. The defense placement taxonomy names this class separately as function return injection, the embedding of malicious instructions within a tool's return payload (Rostamzadeh et al., 2026). The distinction is not pedantic. It determines where a defense would have to sit, and it determines whether the existing body of MCP mitigations reaches this case at all.

It mostly does not, and the literature says so in its own terms. Work on implicit tool poisoning reports that prior indirect injection methods are largely ineffective when their instructions are placed in tool descriptions rather than execution results, with a steep decline in attack success rates (Li et al., 2026). The implication runs against the field's own center of gravity. The vector that receives the most attention, descriptor poisoning, is by that account the weaker one, and the vector that carries instructions in returned results is stronger and comparatively underexamined. A defense coverage analysis reaches a compatible conclusion from the mitigation side, finding protection unevenly distributed and predominantly tool centric, with persistent gaps at the host orchestration, transport, and supply chain

layers, and concluding that many MCP security weaknesses follow from architectural misalignment rather than isolated implementation flaws (Rostamzadeh et al., 2026).

This specimen sits in that gap. It uses the stronger and less studied vector, it runs in production rather than in a benchmark, and it runs through a connector the platform has verified. What the literature has established at scale, through benchmarks against live servers and constructed malicious tools, is that the attack works. What it has not documented is a listed, badged, first party connector doing it to ordinary users for commercial gain. The contribution of this paper is therefore not the discovery of the attack class, which is well described, but the documentation of the class occurring in a verified channel and the governance analysis of why the verification did not stop it.

B. The mechanism in this case

The present case is a direct instance. The connector controls the content of its own tool results. Under the directory policy, a developer must verify that it owns or controls the resources its software returns or renders (Anthropic, 2026b), so the injected instruction is, by the developer's own attestation, the developer's controlled output. That output contained an imperative aimed at the model. The model incorporated the tool result, could not separate the imperative from the surrounding data, and carried out the imperative. This is the mechanism Greshake et al. described, operating exactly as described.

Two design features of the payload deserve attention because they bear on whether the conduct is inadvertent. The first is the camouflage. Injection payloads in the benchmark literature typically announce themselves as instructions, through fake system notices or override directives, rather than blend into the surrounding data (Weng et al., 2026). This payload does both at once. It announces itself in an imperative voice, and it camouflages the announcement by placing the operative command inside a block introduced as ordinary formatting guidance, under a shared heading, in a shared imperative register. By the time the advertising command appears, the block has already been established as routine instruction. The command does not name its own character. It says include the above message, and the phrase the above message performs the concealment, because it refers to promotional text without identifying it as promotional. The second feature is the concealment architecture of the channel itself. Tool results are not surfaced to the user. The user sees the assistant's reply, not the payload that shaped it. The advertisement therefore appears as the terminal content of the assistant's own reply, in the assistant's voice, on a surface the user has been told carries no advertising, while the instruction that produced it remains in a layer the user never sees.

The combination is what makes the exploit effective rather than merely present. A user encounters the advertisement as something the assistant said. Recognizing it as an injected instruction requires the user to know that connectors return payloads, that payloads can contain instructions to the model, and that the model can be made to act on them. The remedy, once recognized, is a single sentence of user instruction, as the specimen shows. The barrier is not the difficulty of the remedy. The barrier is that the user has to know what indirect prompt injection is in order to know that a remedy is available.

V. What the Connector Agreed To

Every listed connector agrees to the directory policy as a condition of listing. The conduct documented above contravenes several of its enumerated provisions. They are presented here in order of strength, because they are not equally firm and the strongest does not depend on characterizing the payload as an advertisement at all.

A. Evasion of model instructions

The policy provides that software must not evade or circumvent the assistant's safety guardrails, system instructions, or sandbox environment (Anthropic, 2026b). The payload's closing line is an instruction directed at the model, commanding output on every turn. Whether the commanded content is an advertisement, a citation format, or anything else is immaterial to this provision. A connector authored an imperative aimed at the model's instruction layer, and the model acted on it. This is the violation in its cleanest form, and it is content neutral.

B. Presentation of instructions

The policy addresses instructions that are hidden or not clearly presented, requiring that any instructions be human readable and clearly presented (Anthropic, 2026b). The command is human readable. The independent requirement is that it be clearly presented, and that phrase must carry meaning beyond human readability or it is redundant with it. The payload's construction cuts against clear presentation. The operative command is nested inside a block introduced as formatting guidance, shares that block's authority, and refers to itself only as the above message. It is legible as text and camouflaged as to its nature. This provision is a supporting count rather than a lead, because a defender may argue that plain English on the page satisfies clear presentation; the counterargument is that presentation reaches the character of the instruction and not only its legibility.

C. Serving advertisements

The policy prohibits software that serves advertisements absent written permission, with the primarily threshold attaching only to the separate promotional vehicle clause (Anthropic, 2026b). Serving one advertisement is within the prohibition. A defender may argue that a first party upsell to the connector's own paid tier is not the third party advertising the February statement of principle chiefly addresses. The argument does not survive three features of the specimen. The prohibition reaches serving advertisements without confining the advertiser to a third party. The same clause separately reaches promotional vehicles, which contemplates promotion generally. And the campaign tracking parameter attached to the pricing link supplies the character directly: an acquisition channel with conversion instrumentation is an advertisement, whatever the relationship between advertiser and product.

D. Token economy and the absence of an option to decline

The policy directs that connectors be frugal with tokens and that, where possible, users be given the option to exclude unnecessary text from the response (Anthropic, 2026b). The advertisement is unnecessary text. It ran by default, on every reply, and no option to exclude it was offered; the only means of stopping it was for the user to recognize it and object. There is a further consequence particular to this arrangement. The reduced result count multiplies the searches a user must run to cover a body of literature, and every search carries the advertisement. The user's context budget, purchased from the platform rather than from the connector, is spent to deliver the connector's advertising, and the free tier's reduction increases how much of that budget the advertising consumes. The cost of the promotion is transferred to the advertised party, denominated in a resource the advertised party bought from a different vendor, on a channel where the party cannot observe the transfer. Cost shifting of this general shape is known in the protocol security literature, which describes a malicious server exploiting the sampling feature to have the user's own model generate content on the server's behalf, moving compute cost to the user (Rostamzadeh et al., 2026). The present case differs in vector and in beneficiary: no capability is abused, the ordinary result path carries the charge, and what is purchased with the user's budget is not computation for the server but advertising aimed back at the user.

The economics of attention make the inversion legible. Formal treatments of attention economies model a fixed population of receivers with finite attention capacity and a set of competing senders who expend resources to capture a share of it; the sender's problem is to buy exposure, and the cost of that exposure is what disciplines how much of it there is (Falkinger, 2007). Attention is the scarce good precisely because it cannot be manufactured at will, is claimed by many parties at once, and is finite for each receiver, which is why so much effort goes into capturing it through targeted advertising, arresting headlines, pop-ups, and, least defensibly, unsolicited messages (Huberman, 2017). The commercial arrangement that ordinarily follows is the one the attention merchant literature describes: a publisher sells its content below cost, and the reader's attention is the product actually sold, with the advertiser as the real customer (Hendricks & Vestergaard, 2018).

The connector channel inverts every term of that arrangement. The reader is not being sold to an advertiser by a publisher who gave them something below cost. The reader purchased the attention surface directly, from a third party, at full price. The context window in which the advertisement appears was bought by the user from the platform, and it is finite in exactly the sense the attention models require. The connector then spends that purchased capacity on its own promotion without buying exposure from anyone, because the ordinary discipline on a sender, that exposure costs the sender money, has been removed. The connector is a sender in Falkinger's sense that pays nothing for radiation. It is not an attention merchant selling an audience it assembled; it is a party appropriating an audience the audience itself paid to convene. Whatever else is true of the arrangement, the party bearing the cost of the advertising is the party the advertising is aimed at, and that party cannot see the transfer, cannot price it, and was not offered the option to decline it.

E. Correspondence of description and behavior

The policy requires that a connector's description precisely match its actual functionality and not include unexpected functionality (Anthropic, 2026b). The connector's documented search response enumerates the fields it returns: the papers, a total count found, and the query (Consensus, 2026b). The documented response does not include an appended promotional message, and it does not include an instruction directed at the model. The documented behavior and the actual behavior differ by one imperative sentence aimed at the assistant, which appears in no published description of the tool.

F. A note on data handling

A secondary compliance question, adjacent to the advertising conduct rather than part of it, concerns data handling. The connector's documentation states that no personal data is stored from its protocol requests (Consensus, 2026b). The same documentation sets the free tier at a fixed number of searches per month, which cannot be metered per account without storing, per account, how many searches an account has run. The connector's privacy policy, in turn, lists the questions and prompts a user submits, together with the output generated, as information it collects, retained as long as reasonably necessary for a legitimate business purpose (Consensus, 2026a). The documentation's assurance and the privacy policy's collection notice cannot both be complete descriptions of what the connector stores. This finding is independent of the advertising and is noted here for completeness; it is developed no further in this paper.

VI. Trust Laundering and the Verification Gap

The specimen will likely be corrected. A single line can be removed from a payload. The finding that outlasts the correction concerns the badge, and it is general to connector directories rather than particular to this connector or this platform.

Call the phenomenon trust laundering: the conversion of a platform's verification mark into credibility for a third party's promotional content, credibility that the content did not earn and that the platform did not knowingly extend. The term is chosen by analogy to money laundering, where the object is to give illegitimate value the appearance of legitimate provenance by passing it through an institution that confers respectability. Here the object passing through is a promotional message, the institution is the directory, and the respectability conferred is the platform's assurance of quality and security. The advertisement does not have to claim the platform endorsed it. It arrives in the assistant's voice, under the platform's badge, on a surface the platform has said is free of advertising, and it inherits the trust attached to all three.

The mechanism that makes trust laundering possible is the badge's semantics. The badge asserts that the platform has reviewed the connector for quality and security. A user reads that as a statement about the connector as it is now: this connector, the one I am about to enable, has been checked and found sound.

That reading is not idiosyncratic, and the trust mark literature suggests it is the expected one. Studying German internet trust marks across a five year interval, Rüdiger and García Rodríguez (2013) found that consumer recognition of a mark and consumer understanding of what the mark certifies come apart and stay apart. Recognition rose over the period; comprehension of the underlying scheme did not follow. The authors identify the resulting misperception as the serious problem, because a consumer who recognizes a mark and trusts it while misconceiving what it stands for makes decisions on a warrant that was never issued. The finding matters here for two reasons. It establishes that the gap between what a mark says and what a user takes it to mean is an empirical regularity rather than a hypothetical, and it locates the harm precisely where this paper locates it: not in a false statement, but in a warranted-seeming statement whose scope the user cannot recover from the mark itself. The directory badge is a strong case of the pattern, because the scope limitation is not merely unstated. It is stated, one line below, in terms that contradict what the mark implies.

The disclaimer on the same card states that the platform cannot verify that listed connectors work as intended or that they will not change. The disclaimer describes exactly what occurred. The connector's behavior is not what a user would take reviewed for quality and security to mean, and the connector's payload can change after review without the review being repeated.

The apparent contradiction resolves into a single finding once the timing is made explicit. A review is an event. It occurs at a moment and produces a result about the artifact as it stood at that moment. A badge that reports the review in the present perfect, with no expiry and no indication that a later change would trigger re-review, presents the result of a momentary event as a standing condition. The platform's own disclaimer concedes the gap: it cannot verify that the artifact will not change, which is to say it cannot guarantee that the reviewed artifact and the present artifact are the same artifact. The badge nonetheless speaks in the register of a present guarantee. The verification asserts a continuing condition that no continuing process secures, and the distance between the asserted condition and the secured one is the room in which trust is laundered.

That the reviewed artifact can mutate after review is not a marginal risk in this setting. A connector's payload is generated server side and can be altered by its developer at any time without touching the listing, the description, or the badge. The review inspects behavior at submission. Nothing in the badge's semantics binds behavior after submission. A directory that issues a present tense assurance over an artifact class defined by post review mutability has built an assurance its own architecture cannot keep.

A. The badge as a signal, and why it does not separate

Signaling theory supplies the vocabulary for what a badge is supposed to do and the precise condition under which it fails. A signal is an observable action that conveys information about an unobservable attribute, and it earns its meaning

through separating equilibrium, the state in which the signal reliably distinguishes high quality senders from low quality ones. The mechanism that produces separation is differential cost. A signal that is costly, but not differentially costly for genuine and imitative senders, does not separate on its own and tends instead toward a pooling equilibrium, in which receivers cannot tell the two apart (Bergh, Connelly, & Ketchen, 2014). The certification literature states the same condition in its own domain: certification provides an effective market signal only where low quality producers face higher certification costs than high quality ones (Anders, Souza Monteiro, & Rouvière, 2010).

Apply that condition to the connector badge. The review, on the evidence of this specimen, does not inspect served payloads for instructions aimed at the model. A connector that injects an advertising command into its tool results and a connector that does not are therefore identical with respect to everything the review examines. Both submit descriptions, both expose schemas, both pass. Obtaining the badge costs the compliant developer and the non compliant developer exactly the same. That is the textbook condition for pooling rather than separation. The badge cannot distinguish the two classes because the attribute distinguishing them is not in the set of things the badge's issuance depends on. What the badge separates is listed connectors from unlisted ones, which is a fact about the directory and not a fact about the software.

There is a second structural problem, distinct from the first. The certification literature is uniform that a third party certifier's credibility depends on its objectiveness and independence from the certified party, and that accreditation by an independent institution is what underwrites the claim to impartiality (Anders et al., 2010). Work on assurance models puts it in terms of role separation: assurance is an intermediary's guarantee of a target's compliance with a standard, and its legitimacy classically depends on degrees of separation between the party setting the rule, the party intermediating, and the party being regulated (Loconto, 2017). The connector directory collapses all three roles into one actor. The platform writes the standard, which is its own directory policy. The platform performs the certification and issues the badge. And the platform is the direct beneficiary of a populous, credible connector ecosystem, because connectors make the assistant more useful and the subscription more valuable. This is not third party certification. It is first party certification of third party software by the party whose product the certification exists to enrich, with no accreditor above it and no separation between the roles that assurance theory requires be kept apart. Neither observation depends on any bad faith by the platform. Both are properties of the arrangement.

B. Post approval mutation, named

The MCP security literature has a name for the attack this gap enables, and the name is instructive. A rug pull is a post approval descriptor mutation that subverts a tool's behavior after it has been reviewed and integrated into normal workflows (Jamshidi et al., 2025; Song et al., 2025). The attack is defined by the interval between approval and mutation, which is precisely the interval the badge's present tense elides. The defense placement analysis states the structural reason directly: the registry layer can only evaluate static metadata at submission time, whereas malicious behavior may only manifest at runtime within the host layer (Rostamzadeh et al., 2026). That is an observation about where defenses can be enforced. This paper draws the normative conclusion that follows from it and that the security literature leaves unstated. If the registry can only evaluate at submission, then a mark issued by the registry can only speak to submission, and a mark that speaks in the present tense about runtime behavior is claiming an authority its own layer does not possess.

None of this is a novel discovery about curated software distribution, which is what makes the badge's confidence harder to defend rather than easier. The federal guidance on vetting the security of mobile applications devotes a section to understanding vetting limitations, on the premise that an organization adopting a vetting process must know in advance what that process cannot establish (Quirolgico et al., 2015). Empirical work on curated app markets reaches the same place from the other direction: strict developer policies coexist with limited knowledge of what the vetting

process actually catches, and apps removed for misbehavior include popular ones that passed review and circulated first (Lin, 2021). Removal after the fact is the observed pattern, which means the interval between review and detection is where the harm lives, and the measured length of that interval in comparable ecosystems is not short. In package registries, where publication proceeds at a rate that outpaces manual inspection, roughly one fifth of identified malicious packages persisted for more than four hundred days before removal (Vu, Pashchenko, & Massacci, 2020). Surveys of those registries describe the same governance shape as the connector directory: publication is open to any verified account, no comprehensive scan gates the upload, users may report malicious code, and the offending artifact continues to do damage until someone takes it down (Kaplan & Qian, 2021). Detection in these ecosystems is a downstream, complaint driven process, and the specimen documented here was found the same way, by a user who noticed. The connector directory inherits every one of these limitations and adds one the mobile case does not have. A mobile binary is a fixed artifact that the store holds a copy of. A connector's served output is not held by anyone but the developer, is regenerated on every call, and can differ from the artifact reviewed without any event the directory would observe. The badge is thus making a stronger claim over a weaker object than the app store precedent it resembles, and the standards literature that precedent produced already counsels the opposite posture: state the limits of the vetting alongside its result.

The specimen also sits within a pattern worth naming precisely, because the pattern is about where this platform's assurances have failed rather than about any single mechanism. In late March 2026 a misconfiguration in the platform's content management system left roughly three thousand unpublished assets, including a draft announcement of an unreleased model, in a publicly accessible store; the platform attributed the exposure to human error in configuration (Fortune, 2026a). In April 2026 the same unreleased model's preview was reported to have been accessed without authorization through one of the platform's third party vendor environments (Fortune, 2026b). These are different mechanisms from the connector injection and from each other, and they should not be conflated with it or with one another as a technical matter. What they share is a location. A platform whose public identity rests on safety and security experienced repeated failures at its third party boundaries: a public data store, a vendor environment, and a verified connector. The advertising injection is the third instance, and it is the one that produces, as its output, the specific thing the platform's advertising commitments promised would not occur.

VII. Deception and the Regulatory Frame

Trust laundering works on the reader because the reader cannot see the advertisement for what it is. Two established bodies of authority describe why that matters and what follows from it.

A. Advertising recognition and persuasion knowledge

The behavioral account begins with the persuasion knowledge model, under which people cope with persuasion attempts by drawing on knowledge of the persuader's tactics and intent, and deploy that knowledge only once they recognize a message as a persuasion attempt (Friestad & Wright, 1994). Recognition is the trigger. The research on native and covert advertising, formats designed to match the surrounding non advertising content, turns on precisely this point: advertising recognition is the key determinant of how consumers respond, and a message that is not recognized as advertising does not activate the skepticism that recognition would bring (Evans & Wojdynski, 2020). When sponsored material is formatted to read as editorial content, readers are less able to identify it as advertising and, when they do, judge it as more deceptive and its source as less credible (Wojdynski, 2016; Campbell & Evans, 2018). The credibility at stake in these studies is not only the advertiser's but the host publication's, which is to say the venue's own trust is spent when it carries advertising the reader cannot recognize (Wojdynski et al., 2018).

The specimen is the limiting case of this literature. In a news publication the sponsored article at least sits on the page, available for inspection, with whatever disclosure the publisher chose to attach. In the connector channel the advertisement arrives in the assistant's own voice, with no disclosure, on a surface the platform has told the user is free of advertising, and the user cannot see the payload that produced it. Recognition, the trigger the whole defense depends on, is denied by the architecture. The reader is not merely undefended. The reader is placed where the defense cannot be raised.

B. Deceptively formatted advertising under Section 5

The regulatory account is the Federal Trade Commission's. Section 5 of the Federal Trade Commission Act prohibits unfair or deceptive acts or practices (15 U.S.C. §45). Under the Commission's Deception Policy Statement, a practice is deceptive where a representation is likely to mislead a reasonable consumer to the consumer's detriment on a material point, and the Commission assesses the overall net impression a representation conveys rather than the literal truth of its individual parts; a practice that misleads a significant minority of reasonable consumers is deceptive (Federal Trade Commission, 1983).

In 2015 the Commission applied these principles to advertising formatted to look like non advertising content. Its Enforcement Policy Statement on Deceptively Formatted Advertisements affirms the long standing principle that advertisements and promotional messages must be identifiable as advertising to consumers, and states that an advertisement is deceptive if it misleads consumers into believing it is something other than an advertisement (Federal Trade Commission, 2015a). The companion business guidance sets out the factors the Commission weighs in assessing net impression, including the advertisement's format, its delivery method, and its target audience, and states the principle that governs this case directly: a misleadingly formatted advertisement is deceptive even when the product claims it communicates are themselves truthful and not misleading (Federal Trade Commission, 2015b).

That principle disposes of the strongest defense available to the connector. A strict reading of the advertisement finds no false sentence; the tool does return ten results and does withhold key takeaways unless the user pays, so the advertisement describes the tool's behavior accurately. Under the Commission's account, literal accuracy is not the question. The question is the net impression on a reasonable consumer in the circumstances of delivery, and a promotional message delivered in the assistant's voice, without any indication that it is a promotional message, on a surface represented as free of advertising, is formatted so as to mislead the consumer about the one thing the format is supposed to make plain: that it is an advertisement at all. The material misimpression is compounded by the sealed channel. The features the advertisement names as paid, a larger result count and key takeaways, are provided without charge by the same vendor on its public website, a fact the in assistant user, who knows the connector only through the assistant, cannot readily discover. The three net impression factors the Commission names all cut the same way here: the format is the assistant's own reply, the delivery method is an instruction the user cannot see, and the target audience is a population sealed off from the vendor's free alternative.

A distinct legal question, which this paper flags rather than resolves, is whether an instruction embedded in a tool result and executed by a model constitutes unauthorized conduct under the Computer Fraud and Abuse Act (18 U.S.C. §1030). The answer is unsettled, and the deception frame is the firmer ground; the point is noted so that later work can take it up rather than assume it.

C. The connector as a credence good

There is a name in economics for a good with the properties this connector has, and it explains why a badge exists at all. Credence goods are those whose quality the non expert consumer cannot verify even after purchase and use, which

distinguishes them from search goods, inspected before buying, and experience goods, assessed after (Darby & Karni, 1973). The seller of a credence good is also the expert who determines how much of the good the buyer needs, and that double role is what generates the incentive to misrepresent (Emons, 2005). The canonical instances are medical, legal, and financial advice and the repair trades, where the layman cannot check the expert's opinion against anything.

The connector is a credence good in the strict sense. The user asks for the literature on a question and receives ten papers. Nothing in the returned set reveals that twenty were found, that the vendor's own website would have shown all twenty free, or that a key takeaway was stripped from each result. The user cannot assess completeness, because assessing completeness is the task they delegated to the tool. Verification would require the user to already possess what they came to acquire. Worse than the canonical case, the connector both supplies the good and defines the need: it decides what the query means, how many results the query merits, and what a result contains, and it does all of this on a channel that shows the user no baseline against which any of it could be compared.

Darby and Karni's structural point is that where quality is unverifiable, the market cannot discipline fraud through inspection and must rely instead on a third party supplying truthful information about quality to the consumer. That is the entire function of a certification mark, and it is why the credence goods literature treats the separation of the expert function into statement and verification as the mechanism that keeps a credence market honest (Thambisetty, 2007). The directory badge is that verification, occupying precisely the institutional slot the theory says a credence market requires. The specimen shows the slot filled by an instrument that does not inspect the attribute in question, issued by a party that is not separate from the market it verifies. Trust laundering, then, is not merely a rhetorical description of what happened. It names the failure of the one mechanism the economics of credence goods says this market cannot function without.

D. Deceptive design beneath the interface

A third body of work locates the specimen more precisely than either of the preceding two. Dark patterns, also called deceptive designs, are commonly defined as design elements that compromise user autonomy by preventing informed choices and that may lead to adverse outcomes for the user (Kollmer & Eckhardt, 2022). Two features of that definition fit the payload exactly. The first is the absence of a decline option: the canonical illustration in this literature is a dialogue that omits the option to refuse, so that autonomy fails for want of a choice rather than for want of information. The advertisement here ran on every reply with no mechanism to exclude it, which is the same structure. The second is that Kollmer and Eckhardt close by anticipating that technological advance would produce novel applications of dark patterns in conversational agents. This specimen is an instance of the class they predicted, arriving through a channel that did not exist when they wrote.

The sharper contribution comes from Leiser and Santos (2023), who argue that deceptive design is inappropriately attributed to the user interface when the pattern is in fact embedded in the system architecture, and who propose a deceptive design visibility spectrum along which patterns range from plainly visible to effectively undetectable. Their title states the thesis: manipulation beneath the interface. Their concern is that a regulatory apparatus trained on interface elements, on buttons and colors and consent walls, will look at the surface and find nothing while the operative design sits in the code below it.

The connector injection is that argument instantiated. There is no interface element to inspect. The user interface displays an ordinary reply from an assistant. Nothing is miscolored, nothing is preselected, no button is missing. The deceptive design is one sentence in a payload the interface never renders, authored by a party the interface never names, executed by a model the user cannot audit. An investigator applying interface based methods to this case would find a clean surface and a compliant looking product. Leiser and Santos predicted that failure mode; the specimen supplies a worked example of it in a channel their paper did not anticipate, which strengthens rather than weakens their claim,

because the visibility spectrum they describe extends further into invisibility than interface bound analysis can reach.

That placement also identifies what an effective remedy would have to inspect. If the pattern lives in the payload, then auditing the interface, the listing, and the description will not find it, and no amount of scrutiny applied to what the user can see will detect what the user cannot. The audit has to read the tool output. Section VI's proposal follows from this: a verification that does not inspect served payloads for instructions directed at the model is verification aimed at the wrong layer.

Finally, Di Porto and Egberts (2023) argue that dark patterns produce harms at a collective as well as an individual level, among them a reduction of trust in markets, and that these collective effects are poorly addressed by laws built around individual transactions. That framing supplies the reason this specimen matters beyond the user who found it. The individual harm is small: an unwanted advertisement, a halved result set, some spent tokens. The collective harm is the depletion of the assurance itself. Every laundered advertisement spends a little of the badge's meaning, and the badge is the instrument on which the whole directory's usefulness rests.

E. A distinct harm to users who rely on the assistant as accommodation

There is a harm that the deception frame does not fully capture. For a user whose reason for using the connector inside the assistant is to avoid the friction of moving between applications, a result cap that exists in the connector and not on the vendor's website converts the tool into a source of the very friction it was adopted to remove. The reduced count compels repeated searches, and the only path to the fuller result set is to leave the assistant and run the search again in the place where the vendor did not reduce it. A tier structure that degrades the in assistant channel specifically imposes its heaviest cost on the users for whom the in assistant channel is not a convenience but an accommodation.[1]

VIII. Conclusion

A verified connector delivered advertising through an assistant whose maker had committed, in public and in policy, that the assistant would not carry advertising. It did so by the most anticipated attack in the category, an instruction hidden in a tool result, on a channel where the user could see neither the instruction nor the option to refuse it. Two parties bear the outcome. The connector's developer authored an instruction it had agreed, as a condition of listing, not to author. The platform verified the connector, issued a present tense assurance of quality and security, and ran a review that did not detect the field's best known failure mode in the payload of a connector it then vouched for.

The advertisement can be removed. What remains after it is removed is the reason it was possible: a verification badge that speaks in the present tense about an artifact its issuer's own disclaimer concedes may have changed since it was reviewed, and a trust that is laundered through that gap. A directory that means to make its badge trustworthy has to close the distance between the two. A verification worth the word would inspect tool output for instructions directed at the model, would re-review on any change to a connector's served behavior, and would speak in a register that matches what it can actually secure. Until it does, the badge certifies a moment and presents a state, and the gap between them is exactly wide enough for an advertisement, or for whatever a payload carries next.

[1] Whether and how to develop this accessibility harm in the first person, including any disclosure of the author's own basis for raising it, is a decision reserved to the author for a later version.

ACKNOWLEDGMENTS

The author discloses the use of speech-to-text software (Wispr Flow) and Anthropic's Claude as assistive technology for a diagnosed neurodevelopmental condition. All intellectual contributions, research questions, theoretical arguments, methodological decisions, and conclusions are solely the author's own.

Correspondence concerning this article should be addressed to Travis Gilly, Real Safety AI Foundation, Illinois, United States. Email: t.gilly@ai-literacy-labs.org.

REFERENCES

Anthropic. (2026a, February 4). *Claude is a space to think*. Retrieved from https://www.anthropic.com/news/claude-is-a-space-to-think

Anthropic. (2026b). *Anthropic software directory policy*. Retrieved from https://support.claude.com/en/articles/13145358-anthropic-software-directory-policy

Anders, S., Souza Monteiro, D. M., & Rouvière, E. (2010). Competition and credibility of private third-party certification in international food supply. *Journal of International Food & Agribusiness Marketing*, 22(3–4), 328–341. https://doi.org/10.1080/08974431003641554

Bergh, D. D., Connelly, B. L., & Ketchen, D. J. (2014). Signalling theory and equilibrium in strategic management research: An assessment and a research agenda. *Journal of Management Studies*, 51(8), 1334–1360. https://doi.org/10.1111/joms.12097

Campbell, C., & Evans, N. J. (2018). The role of a companion banner and sponsorship transparency in recognizing and evaluating article-style native advertising. *Journal of Interactive Marketing*, 43, 17–32.

Consensus. (2026a, May 15). *Privacy policy*. Retrieved from https://consensus.app/home/privacy-policy/

Consensus. (2026b). *Getting started with the Consensus MCP*. Retrieved from https://docs.consensus.app/docs/mcp

Darby, M. R., & Karni, E. (1973). Free competition and the optimal amount of fraud. *The Journal of Law and Economics*, 16(1), 67–88. https://doi.org/10.1086/466756

Di Porto, F., & Egberts, A. (2023). The collective welfare dimension of dark patterns regulation. *European Law Journal*, 29(1–2), 114–141. https://doi.org/10.1111/eulj.12478

Emons, W. (2005). Credence goods: The monopoly case. In *Economics of information* (pp. 17–42). De Gruyter. https://doi.org/10.1515/9783110508123-003

Evans, N. J., & Wojdynski, B. W. (2020). An introduction to the special issue on native and covert advertising formats. *International Journal of Advertising*, 39(1), 1–3.

Falkinger, J. (2007). Attention economies. *Journal of Economic Theory*, 133(1), 266–294. https://doi.org/10.1016/j.jet.2005.12.001

Federal Trade Commission. (1983). *Policy statement on deception* (appended to Cliffdale Associates, Inc., 103 F.T.C. 110, 174 (1984)). Retrieved from https://www.ftc.gov/system/files/documents/public_statements/410531/831014deceptionstmt.pdf

Federal Trade Commission. (2015a). *Enforcement policy statement on deceptively formatted advertisements*. Retrieved

from https://www.ftc.gov/system/files/documents/public_statements/896923/151222deceptiveenforcement.pdf

Federal Trade Commission. (2015b). *Native advertising: A guide for business*. Retrieved from https://www.ftc.gov/business-guidance/resources/native-advertising-guide-businesses

Fortune. (2026a, March 26). Anthropic says it is testing Mythos, a powerful new AI model, after a data leak reveals its existence. *Fortune*. Retrieved from https://fortune.com/2026/03/26/anthropic-says-testing-mythos-powerful-new-ai-model-after-data-leak-reveals-its-existence-step-change-in-capabilities/

Fortune. (2026b, April 23). A group of users leaked Anthropic's AI model Mythos by reportedly guessing where it was located. *Fortune*. Retrieved from https://fortune.com/2026/04/23/anthropic-mythos-leak-dario-amodei-ceo-cybersecurity-hackers-exploits-ai/

Friestad, M., & Wright, P. (1994). The persuasion knowledge model: How people cope with persuasion attempts. *Journal of Consumer Research*, 21(1), 1–31.

Greshake, K., Abdelnabi, S., Mishra, S., Endres, C., Holz, T., & Fritz, M. (2023). Not what you've signed up for: Compromising real-world LLM-integrated applications with indirect prompt injection. In *Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security* (AISec '23). Association for Computing Machinery. https://doi.org/10.1145/3605764.3623985

Hendricks, V. F., & Vestergaard, M. (2018). The attention economy. In *Reality lost: Markets of attention, misinformation and manipulation* (pp. 1–17). Springer. https://doi.org/10.1007/978-3-030-00813-0_1

Huberman, B. A. (2017). Big data and the attention economy. *Ubiquity*, 2017(December), 1–7. https://doi.org/10.1145/3158337

Hou, X., Wang, S., & Zhang, Y. (2026). SMCP: Secure Model Context Protocol. *arXiv* preprint arXiv:2602.01129. https://doi.org/10.48550/arxiv.2602.01129

Huang, C., Huang, X., & Tran, N. P. (2026). Model Context Protocol threat modeling and analyzing vulnerabilities to prompt injection with tool poisoning. *arXiv* preprint arXiv:2603.22489. https://doi.org/10.48550/arxiv.2603.22489

Jamshidi, S., Nafi, K. W., & Moradi Dakhel, A. (2025). Securing the Model Context Protocol: Defending LLMs against tool poisoning and adversarial attacks. *arXiv* preprint arXiv:2512.06556. https://doi.org/10.48550/arxiv.2512.06556

Kollmer, T., & Eckhardt, A. (2022). Dark patterns: Conceptualization and future research directions. *Business & Information Systems Engineering*, 65(2), 201–208. https://doi.org/10.1007/s12599-022-00783-7

Leiser, M., & Santos, C. (2023). *Dark patterns, enforcement, and the emerging digital design acquis: Manipulation beneath the interface*. Center for Open Science. https://doi.org/10.31235/osf.io/rf3ja

Lin, F. (2021). Demystifying removed apps in iOS app store. *arXiv* preprint arXiv:2101.05100. https://doi.org/10.48550/arxiv.2101.05100

Kaplan, B., & Qian, J. (2021). A survey on common threats in npm and PyPi registries. *arXiv* preprint arXiv:2108.09576. https://doi.org/10.48550/arxiv.2108.09576

Li, R., Wang, Z., & Yao, Y. (2026). MCP-ITP: An automated framework for implicit tool poisoning in MCP. *arXiv* preprint arXiv:2601.07395. https://doi.org/10.48550/arxiv.2601.07395

Loconto, A. M. (2017). Models of assurance. *The Annals of the American Academy of Political and Social Science*, 670(1), 112–132. https://doi.org/10.1177/0002716217692517

Model Context Protocol. (2025). *Specification (2025-11-25)*. The Linux Foundation. Retrieved from https://modelcontextprotocol.io/specification/2025-11-25

OWASP. (2025). *OWASP Top 10 for LLM applications 2025: LLM01 prompt injection*. Open Worldwide Application Security Project. Retrieved from https://genai.owasp.org/llmrisk/llm01-prompt-injection/

Perez, F., & Ribeiro, I. (2022). Ignore previous prompt: Attack techniques for language models. *arXiv* preprint arXiv:2211.09527. https://doi.org/10.48550/arxiv.2211.09527

Quirolgico, S., Voas, J., & Karygiannis, T. (2015). *Vetting the security of mobile applications* (NIST Special Publication 800-163). National Institute of Standards and Technology. https://doi.org/10.6028/nist.sp.800-163

Rostamzadeh, M., Narula, S., & Birhan, N. (2026). MCP-DPT: A defense-placement taxonomy and coverage analysis for Model Context Protocol security. *arXiv* preprint arXiv:2604.07551. https://doi.org/10.48550/arxiv.2604.07551

Rüdiger, K., & García Rodríguez, M. J. (2013). Do we need innovative trust intermediaries in the digital economy? *Global Business Perspectives*, 1(4), 329–340. https://doi.org/10.1007/s40196-013-0021-8

Song, X., et al. (2025). *Beyond the protocol: Unveiling attack vectors in the Model Context Protocol ecosystem*. *arXiv* preprint arXiv:2506.02040. https://doi.org/10.48550/arxiv.2506.02040

Thambisetty, S. (2007). Patents as credence goods. *Oxford Journal of Legal Studies*, 27(4), 707–740. https://doi.org/10.1093/ojls/gqm021

Vu, D.-L., Pashchenko, I., & Massacci, F. (2020). Towards using source code repositories to identify software supply chain attacks. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security* (pp. 2093–2095). https://doi.org/10.1145/3372297.3420015

Wang, Y.-M., Chen, S., Alkhudair, R., et al. (2025). Defending against prompt injection with DataFilter. *arXiv* preprint arXiv:2510.19207. https://doi.org/10.48550/arxiv.2510.19207

Weng, S., Feng, Y., & Zhang, J. (2026). ARGUS: Defending LLM agents against context-aware prompt injection. *arXiv* preprint arXiv:2605.03378. https://doi.org/10.48550/arxiv.2605.03378

Wojdynski, B. W. (2016). The deceptiveness of sponsored news articles: How readers recognize and perceive native advertising. *American Behavioral Scientist*, 60(12), 1475–1491.

Wojdynski, B. W., Evans, N. J., & Hoy, M. G. (2018). Measuring sponsorship transparency in the age of native advertising. *Journal of Consumer Affairs*, 52(1), 115–137.

Yusifov, T., Aghayev, A., & Hasanzada, L. (2026). Taxonomy of prompt injection attacks and analysis of defense mechanisms in large language model-based chatbots. *InterConf*, 69(295), 161–176. https://doi.org/10.51582/interconf.19-20.05.2026.017