Many Looks Beat One Look: Why Full Context Does Not Help Commercial LLM Guardrails Against Multi-Turn Attacks

Manav Thakkar
Independent Researcher
Dallas, United States
manavsutd@gmail.com

***Abstract*—Enterprises put a commercial guardrail in front of their language models and show it one message at a time, as vendor documentation instructs. Multi-turn attacks defeat this, and the obvious remedy is to show it the whole conversation. We test that remedy on four commercial guardrails (AWS Bedrock Guardrails, Azure Prompt Shields, Azure Content Safety, and Google Vertex Model Armor) over 2,489 attack and 1,227 benign multi-turn conversations. In aggregate, no: conversation-level detection is no higher under full context for any product at any filter strength, and falls for two. Yet on a single decision, every product that detects anything detects more when shown the conversation, Bedrock rising from 38 to 56 percent. The turn-by-turn default wins in aggregate because it takes many weakly correlated looks whose union beats any single look, whereas nested full-context prefixes are strongly correlated. A prefix that fired stops firing once extended in 8 to 28 percent of flagged conversations, and a third to a half of turn-by-turn detections land on the first turn. Full context hurts most on topic drift, where benign role-play outweighs the final harmful request. One detection-only pass that logs every graded internal detector reconstructs the whole strength curve and shows which harm families evade every filter. Thirteen percent of attacks pass all four products. We release the harness.**

***Keywords*—AI security, large language models, guardrails, content moderation, multi-turn jailbreaks, security measurement, prompt injection**

I. Introduction

Organisations that deploy large language models (LLMs) rarely rely on model alignment alone. They buy a separate guardrail service—AWS Bedrock Guardrails, Azure AI Content Safety, Google Vertex AI Model Armor—and place it in front of the model. In regulated settings it is the control an auditor points to.

Multi-turn attacks reliably defeat LLM safety. An attacker escalates gradually [1], drifts across topics [3], or decomposes a harmful objective into individually benign requests [9]. The prevailing explanation is that guardrail classifiers are stateless and cannot perceive intent spread across turns; the prevailing remedy is to show them the conversation. This paper tests both on the deployed products: the explanation misdiagnoses the failure and the remedy does not work, for a reason not measured before.

A deployed guardrail is a classifier plus a decision about *what text it is handed*, and that decision—made by the integrator, following the vendor's own documentation—is almost always to hand it the latest message. Vertex Model Armor says it inspects each prompt independently, does not track conversation history, and should be sent only the latest message [20]. Bedrock's conversation guidance evaluates only the most recent message [18]. Prompt Shields takes a single user prompt plus a handful of documents [19]. A control that never sees the conversation ought to miss attacks spread across it. That expectation has been tested by toggling history on an open framework [28], never on the deployed commercial services at their documented configurations, and never with per-detector attribution. On those services, the two levels of the question disagree. Per decision, every product that detects anything uses context: shown the whole conversation, it detects more. Per conversation, where the guardrail fires if it flags any turn, showing the whole conversation never helps and for two products hurts. The turn-by-turn default wins not by perceiving escalation but by sampling: it takes many weakly correlated looks, and their union beats one better look.

Three questions organise the measurement: *(i)* how much of the multi-turn miss rate is a deployment choice rather than a classifier limit; *(ii)* at which level of accounting—per decision or per conversation—showing the classifier the whole conversation helps, and why the levels disagree; *(iii)* which internal detectors fail when a product misses an attack, and whether some harm families fall through all of them. The third requires opening the black box: we run each product in detection-only mode and capture every internal filter's decision and confidence on every call, which also lets one pass reconstruct behaviour at every filter strength.

Our contributions are:

- The mechanism behind the multi-turn gap on deployed guardrails: on four commercial services, at every reconstructed strength, conversation-level detection is never higher under full context although, as run, per-decision detection is higher for every product that detects anything. Nested prefixes are strongly correlated and "un-fire"; the effect is family-specific, not a length artefact.
- A decomposition of the multi-turn miss rate into configuration-induced and model-induced components, with two propositions on when bounded inspection provably fails, including a condition (Proposition 2) under

XXX-X-XXXX-XXXX-X/XX/$XX.00 © 20XX IEEE

which padding restores the miss at any input limit.
- A detection-only measurement methodology in which a single pass reconstructs a product's full strength-threshold operating curve, and a *per-detector* view of which internal filters carry multi-turn detection and which harm families evade all of them.
- An inspection-surface taxonomy of commercial guardrails from vendor documentation and measured behaviour, including whether each service returns per-detector confidence (Table I).
- A released, vendor-agnostic harness[1] with per-product configuration guidance and a stateful adapter that accumulates per-turn scores under a fixed false-positive budget; a reference tier of three open guard models for context.

II. What Commercial Guardrails Inspect

Table I records what each service scores, from vendor documentation [18]–[20] confirmed against observed responses. Four patterns recur: no service keeps safety state across calls (history is re-sent, and re-billed, by the caller); several provide no field for history at all; every service imposes a finite input limit (Section V-F). Central to our method, most detectors return a graded confidence rather than a bare block decision, so one detection-only pass at maximum sensitivity exposes the full operating curve. The exception, Prompt Shields, returns a binary verdict and admits no reconstruction.

The "Level" column records the state available to a single decision: L0 scores the current turn only; L1 accepts caller-supplied history but labels the last turn. Every deployed product we study operates at L0 or, when the integrator resends history, L1; none keeps session state of its own.

III. Threat Model and Measurement Model

A. Definitions

A conversation is a sequence of user turns $u_1, \ldots, u_T$ and assistant turns $a_1, \ldots, a_T$; the prefix at step t is $P_t = (u_1, a_1, \ldots, u_t)$. A *detector* D maps a scored unit to a risk score and fires when the score reaches a threshold τ. An *inspection function* I_c maps P_t to the unit actually submitted to D under configuration c. The deployed decision at turn t is $D(I_c(P_t))$, and a conversation is detected if the guardrail fires at any turn:

$$\mathrm{det}_c(C) = \max_{1 \le t \le T} \mathbf{1}[D(I_c(P_t)) \ge \tau]\,. \tag{1}$$

Over an attack corpus $\mathcal{A}$ the conversation detection rate is $\mathrm{CDR}_c = \frac{1}{|\mathcal{A}|} \sum_{C \in \mathcal{A}} \mathrm{det}_c(C)$, and the guardrail multi-turn gap between two configurations is $\mathrm{GMG}(c_1, c_2) = \mathrm{CDR}_{c_2} - \mathrm{CDR}_{c_1}$.

[1] https://github.com/Manav-claude/guardrail-gap

B. Decomposing the miss rate

Under the deployed default $c = \mathsf{last}$, the miss rate splits exactly:

$$\underbrace{1 - \mathrm{CDR}_{\mathsf{last}}}_{\text{observed miss}} = \underbrace{(\mathrm{CDR}_{\mathsf{full}} - \mathrm{CDR}_{\mathsf{last}})}_{\text{configuration-induced}} + \underbrace{(1 - \mathrm{CDR}_{\mathsf{full}})}_{\text{model-induced}}. \tag{2}$$

The first term is recoverable by a deployment change; the second is a property of the classifier given the whole prefix. Reporting only the left-hand side, as prior work does, conflates a procurement problem with a research problem. The configuration-induced term *need not be positive*: if per-turn looks are less correlated than nested prefixes, their union can beat any single full look, and the term is zero or negative—the outcome for every detector we test (Section V-B).

C. Threshold reconstruction

Running D in detection-only mode at its most sensitive setting and recording the graded confidence $\gamma \in \{\ell_1 < \cdots < \ell_m\}$ of each internal detector lets us recover the decision at *any* configured strength without re-querying: a strength that acts at level ℓ_j fires exactly when $\gamma \ge \ell_j$. One pass therefore yields the entire operating curve, which is what makes a per-detector, multi-strength study affordable.

D. When bounded inspection provably fails

Call I_c *k-bounded* if $I_c(P_t)$ depends only on the last k turns or tokens of P_t; say C admits a *k-benign decomposition* w.r.t. D if every window of at most k turns scores below τ.

Proposition 1: If I_c is k-bounded and C admits a k-benign decomposition w.r.t. D, then $\mathrm{det}_c(C) = 0$, whatever the accuracy of D on longer units.

Proof. Every $I_c(P_t)$ is a window of at most k turns and scores below τ by hypothesis, so no turn fires and the maximum in (1) is zero. □

Proposition 2: Let D have input limit κ and let C be detected under full. If benign filler can be inserted so that every κ-sized window of the padded conversation scores below τ, the padded conversation admits a κ-benign decomposition and is missed, by Proposition 1 with $k = \kappa$.

Proposition 2 is why no wider window is a durable fix either: every service in Table I has a finite limit (Section V-F). The complementary regime, in which an attacker splits work across unlinkable sessions or accounts, is settled negatively elsewhere [12] and inherited here.

IV. Method

A. Guardrails and detection-only configuration

We evaluate four commercial services: AWS Bedrock Guardrails (its six content filters plus eight denied topics we defined, since Bedrock ships no default topic list; definitions are in the artifact), Azure Prompt Shields, Azure Content Safety text analysis (eight-level severity), and Google Vertex Model Armor (responsible-AI filters, prompt-injection/jailbreak, malicious-URI, CSAM). Each is configured with every filter at maximum sensitivity and, where supported, in *detection-only* mode, which reports findings without

TABLE I

INSPECTION SURFACE OF THE GUARDRAILS STUDIED, FROM VENDOR DOCUMENTATION (ACCESSED 2026) AND CONFIRMED BY OBSERVED RESPONSES. "OWN STATE": SERVER-SIDE SAFETY STATE ACROSS CALLS; "PER-DETECTOR CONF.": THE RESPONSE GRADES EACH INTERNAL FILTER (ENABLING THRESHOLD RECONSTRUCTION).

Service	Scored unit	History	Own state	Per-detector conf.	Documented limit	Level
AWS Bedrock Guardrails	content blocks	caller-supplied, re-billed	no	yes (4 levels)	text unit = 1000 chars	L0–L1
Azure Prompt Shields	`userPrompt`+docs	no history field	no	**no** (binary)	10K chars; $\leq$5 docs	L0
Azure Content Safety (text)	text	no	no	yes (0–7 severity)	10K chars	L0
Google Vertex Model Armor	single prompt/response	discouraged	no	yes (3 levels)	per-token; latest msg	L0
Reference tier (open models, run locally; not commercial services)						
Meta Prompt Guard 2	single text	no	no	yes (probability)	512 tokens	L0
ProtectAI DeBERTa v2	single text	no	no	yes (probability)	512 tokens	L0
Qwen3Guard-0.6B	prompt or conversation	if supplied	no	yes (3 classes)	model context	L0–L1

blocking: detection is identical to a blocking deployment while every internal detector is exposed. Guardrail versions are pinned, every raw response (per-detector decision, confidence, service-reported billing units, latency, and inspected-character coverage) is persisted and runs are date-stamped. Three open guard models—Meta Prompt Guard 2, ProtectAI DeBERTa v2, and Qwen3Guard-0.6B—form a reference tier run locally; they are open detectors, not commercial services, and are reported separately and never merged into the commercial results.

B. Configurations and corpora

Each service is run under three inspection configurations: last (submit u_t only; the vendor-documented pattern), full (submit the user turns $u_1 \| \cdots \| u_t$), and m2s (the multi-turn-to-single-turn *hyphenize* template [11]: the same user turns rendered as one bulleted list, so a single-prompt detector sees the history as one structured unit). Attacks are drawn from three families: gradual escalation, using human multi-turn jailbreaks [2]; topic drift, using actor-driven dialogues [3] whose harmful request arrives only at the final turn; and coreference, using CoSafe [6], where a harmful antecedent introduced early is referenced pronominally later so the final turn is underspecified on its own. Decomposition into individually benign requests [9] is not covered. Benign conversations come from WildChat [21] and MT-Bench-101 [22], filtered to remove flagged content; their mean length matches the attacks (4.0 turns each). We use 2489 attack conversations (489 escalation, 600 drift, 1400 coreference) and 1227 benign conversations, each with its dataset harm category. No model is prompted to generate harmful content at any point: the corpora are pre-existing and labelled, and only the guardrail classifier is invoked.

C. Metrics

We report conversation-level detection CDR, the first turn at which a product fires, the conversation-level false-positive rate FPR on benign data, latency, and cost from service-reported billing units. Wilson intervals for every rate are released with the artifact (half-widths are at most 0.045 for the smallest family). Because configurations are evaluated on identical conversations, pairwise comparisons use McNemar's test with continuity correction, and Cochran's Q with Holm-corrected pairwise tests across the three configurations. An internal detector is credited with a conversation when it fires on any turn under the given configuration; we report each detector's detection rate by family and its benign false-positive rate.

V. RESULTS

A. The configuration gap, per product

Table II reports detection by service, family and configuration. Under the deployed last default, detection is uneven: Model Armor and Bedrock catch most escalation attacks (0.90 and 0.82), Azure Text a quarter, Prompt Shields essentially none. Topic drift is the hardest family for every product, and 13% of attacks (329 of 2489; 24% of drift) pass all four products under last, 12% under every configuration. The conversation-level gap is never positive: Prompt Shields and Azure Text are flat (GMG $= 0.000$, $p = 1$ and $p = 0.93$), while Bedrock and Model Armor fall in aggregate—GMG(last, full) $= -0.079$ (p=2.4×10^{-27}) and -0.143 (p=3.3×10^{-66}), $n = 2489$. Cochran's Q over the three configurations agrees, and m2s is indistinguishable from full for Bedrock and both Azure services (Holm p>0.1). The decomposition (2) attributes the miss to the model-induced term (Bedrock 0.41, Model Armor 0.36, Prompt Shields 1.00, Azure Text 0.89). Two operating-point failures stand out: Model Armor flags 42% of benign conversations at its most sensitive setting, and Prompt Shields—a jailbreak/injection detector—has AUROC 0.49 on these harm-oriented attacks, indistinguishable from chance.

B. Many looks beat one look

The aggregate drop does not mean context is ignored (Table III). Scored on a single decision, the final turn alone, every product that detects anything detects *more* when shown the whole conversation: Bedrock $0.38 \to 0.56$ (its single-decision FPR also rises, $0.06 \to 0.11$), Model Armor $0.41 \to 0.54$ with FPR falling $0.21 \to 0.13$, Azure Text $0.03 \to 0.09$ (all p<10^{-25}). The turn-by-turn default wins in aggregate because it takes T weakly correlated looks whose union exceeds any single decision, whereas nested prefixes are strongly correlated and, in 8–28% of conversations that ever fire, a prefix that

TABLE II

COMMERCIAL GUARDRAILS: CONVERSATION-LEVEL DETECTION RATE BY FAMILY AND INSPECTION CONFIGURATION AT MAXIMUM SENSITIVITY; BENIGN FALSE-POSITIVE RATE IN THE LAST COLUMN. DETECTION IS CREDITED IF ANY DETECTOR FIRES ON ANY TURN. M2S ROWS (WITHIN 0.05 OF FULL) AND WILSON INTERVALS ARE IN THE ARTIFACT.

Service	Config	Escal.	Drift	Coref.	FPR
Bedrock	last	0.82	0.62	0.64	0.14
	full	0.82	0.50	0.55	0.13
Prompt Shields	last	0.00	0.00	0.00	0.01
	full	0.00	0.00	0.00	0.01
Azure Text	last	0.24	0.02	0.11	0.02
	full	0.24	0.02	0.11	0.02
Model Armor	last	0.90	0.54	0.85	0.42
	full	0.76	0.26	0.76	0.22

TABLE III

AGGREGATE (ANY TURN) VERSUS SINGLE-DECISION (FINAL TURN ONLY) DETECTION UNDER LAST AND FULL, WITH THE BENIGN FALSE-POSITIVE RATE OF THE SINGLE DECISION. WHERE A PRODUCT DETECTS ANYTHING, FULL CONTEXT WINS THE SINGLE DECISION BUT NEVER THE AGGREGATE.

	Any turn		Final turn		Final-turn FPR	
Detector	last	full	last	full	last	full
Bedrock	0.67	0.59	0.38	0.56	0.06	0.11
Prompt Shields	0.00	0.00	0.00	0.00	0.00	0.01
Azure Text	0.11	0.11	0.03	0.09	0.01	0.01
Model Armor	0.78	0.64	0.41	0.54	0.21	0.13
Qwen3Guard (ref.)	0.33	0.29	0.10	0.26	0.01	0.02

fired stops firing when extended. Indeed, 32–46% of last detections occur on the first turn, before any escalation or coreference exists. The effect is family-specific, not a length artefact: within escalation, the only family with wide length variation (2–34 turns), the gap is flat in length (Spearman $\rho \approx 0$, all products). With Bedrock's vendor content filters alone, escalation detection *rises* under full ($0.68 \rightarrow 0.74$, $p = 4 \times 10^{-4}$) while drift collapses ($0.41 \rightarrow 0.20$, $p < 10^{-25}$): the harmful final request is outweighed by the benign role-play before it. For Model Armor the aggregate drop is largely an operating-point shift, since FPR halves under full and AUROC is unchanged (0.72 vs. 0.73).

C. At other strengths

Because every response grades each detector, the whole strength curve follows from the one pass (Section III). At Bedrock's MEDIUM and LOW strengths, last detection falls to 0.62 and 0.52 (FPR 0.11 and 0.08) and the gap stays negative (-0.09 at both, $p < 10^{-31}$). Model Armor at its default MEDIUM_AND_ABOVE detects 0.29 of attacks (escalation 0.35, drift 0.05) at FPR 0.044, and 0.07 at HIGH; its gap shrinks to -0.02 ($p = 0.01$) and then to zero. Azure Text at its documented default (severity ≥ 4) detects 0.035 at FPR 0.006 (Table IV). Maximum sensitivity thus overstates what a default deployment catches about threefold, and no strength makes the gap positive.

TABLE IV

OPERATING POINTS RECONSTRUCTED FROM THE SINGLE DETECTION-ONLY PASS: POOLED ATTACK DETECTION AND BENIGN FPR UNDER LAST AND FULL AT EACH FILTER STRENGTH. BEDROCK ROWS INCLUDE THE DENIED TOPICS.

Service	Strength	last		full		GMG
		CDR	FPR	CDR	FPR	
Bedrock	HIGH (run)	0.67	0.14	0.59	0.13	−0.08
	MEDIUM	0.62	0.11	0.53	0.10	−0.09
	LOW	0.52	0.08	0.43	0.07	−0.09
Model Armor	LOW+ (run)	0.78	0.42	0.64	0.22	−0.14
	MEDIUM+	0.29	0.04	0.27	0.03	−0.02
	HIGH	0.07	0.02	0.07	0.02	−0.00
Azure Text	sev. ≥ 2 (run)	0.11	0.02	0.11	0.02	0.00
	sev. ≥ 4	0.04	0.01	0.04	0.00	0.00
	sev. ≥ 6	0.01	0.00	0.01	0.00	0.00

TABLE V

PER-DETECTOR VIEW (BEDROCK, LAST CONFIGURATION): DETECTION RATE OF EACH INTERNAL FILTER BY HARM FAMILY; BENIGN FPR IN THE LAST COLUMN. ROWS ARE THE SIX VENDOR CONTENT FILTERS AND THE EIGHT DENIED TOPICS WE DEFINED; MERGED ROWS COUNT A CONVERSATION WHEN ANY LISTED DETECTOR FIRES.

Internal detector	Escal.	Drift	Coref.	FPR
content: Prompt-attack	0.49	0.00	0.01	0.10
content: Misconduct	0.20	0.33	0.20	0.02
content: Violence	0.13	0.04	0.11	0.02
content: Hate/Insults/Sexual	0.14	0.04	0.20	0.04
topic: Cyber intrusion	0.15	0.23	0.07	0.01
topic: Weapons / CBRN	0.09	0.09	0.09	0.01
topic: Fraud / Financial	0.26	0.14	0.14	0.01
topic: Terrorism/Drugs/other	0.33	0.31	0.28	0.02

D. Opening the black box: per detector

Table V breaks one product down to its internal detectors: each row is a filter, each column a harm family, each cell the fraction of attacks in that family the filter flags, with the benign false-positive rate. This exposes division of labour and blind spots—which harm families evade *every* detector at once—that the aggregate rate hides. Bedrock's escalation detection leans most heavily on the prompt-attack filter (0.49), which contributes a 0.10 benign false-positive rate; the topic filters add targeted coverage (fraud 0.26 on escalation, cyber intrusion 0.23 on drift). Yet on the 84 drift conversations that explicitly solicit chemical or biological harm, the dedicated Weapons/CBRN topic fires on 0.36 and all fourteen detectors together on 0.62: more than a third pass every filter. Model Armor's responsible-AI filters carry its detection (dangerous 0.68 on escalation) but each is a major false-positive source (dangerous alone flags 0.25 of benign input). Equivalent per-detector tables for the other products are released with the artifact.

E. What context costs

Full-context inspection raised measured billing units by $1.29\times$ over last for Bedrock and $1.26\times$ for the Azure services (Model Armor bills per request). This is well below the quadratic worst case because the conversations are short (mean

TABLE VI
REFERENCE TIER (OPEN GUARD MODELS, FULL CORPUS n=2489 ATTACKS / 1227 BENIGN): CONVERSATION-LEVEL DETECTION RATE BY HARM FAMILY AND INSPECTION CONFIGURATION; BENIGN FPR IN THE LAST COLUMN. OPEN DETECTORS, NOT COMMERCIAL SERVICES; NOT COMPARABLE TO TABLE II.

Model	Config	Escal.	Drift	Coref.	FPR
Prompt Guard 2	last	0.07	0.04	0.01	0.02
	full	0.05	0.01	0.00	0.02
ProtectAI v2	last	0.43	0.07	0.10	0.11
	full	0.10	0.01	0.10	0.07
Qwen3Guard	last	0.37	0.12	0.40	0.03
	full	0.23	0.06	0.42	0.02

4.0 turns); re-sending the prefix grows as $O(T^2)$ in the long sessions typical of agentic deployments. Truncation was rare—Bedrock and Model Armor never reported coverage below one, the Azure services on 0.6% of full calls—so the aggregate drop is not lost text. Median latency per call was 0.2 s (Bedrock, Model Armor) to 0.6 s (Azure) in every configuration.

F. Dilution

Proposition 2 states that padding a detected conversation with benign filler until no κ-sized window scores above τ defeats detection. Every product in Table I has a finite limit (512 tokens for Prompt Guard 2; 10,000 characters for the Azure services), so the attack is available against all of them in principle. A padding sweep is the next study with the released harness. The drift result already shows dilution below the limit: proportional dilution of the payload by benign context, measured for open guardrails in [27], appears here on commercial APIs.

G. Reference tier: open guard models

To place the commercial results in context we run three open guard models over the entire corpus: Prompt Guard 2 [24] and ProtectAI DeBERTa v2[2] (prompt-injection detectors) and Qwen3Guard-0.6B [23] (a harm classifier). They are open detectors, not deployed services, and are reported separately (Table VI). Even the strongest, Qwen3Guard, separates attacks from benign conversations only moderately (AUROC 0.81) and misses three in five escalation attacks. ProtectAI is at chance on these harm-oriented attacks (AUROC 0.51), a reminder that a guardrail's advertised category and its effective coverage differ. All three show the same negative pooled gap ($\mathrm{GMG}(\mathsf{last}, \mathsf{full})$ of -0.015, -0.080 and -0.032; all $p<10^{-4}$). Qwen3Guard also shows the positive per-decision gap ($0.10 \to 0.26$); the two injection detectors, off-target to begin with, do not ($0.10 \to 0.07$ and $0.01 \to 0.01$). The mechanism is a property of harm classifiers under turn-by-turn scoring, not of the commercial services alone.

H. A stateful adapter

Wrapping each unmodified service in a cumulative-sum detector over its own per-turn scores ($S_t = \max(0, S_{t-1} + r_t - \delta)$, alarm when $S_t > h$, with (δ, h) chosen on the benign set for a 5% false-positive budget) adds one scalar of session state at no API cost. Within the budget Azure Text rises from 0.11 to 0.23 and Qwen3Guard from 0.33 to 0.45, and Bedrock obtains its only operating point under 5% FPR (0.37 at 0.046; no per-turn strength reaches one). Model Armor gains nothing and the injection detectors lose, and the gain shrinks with the budget (Azure Text $+0.02$ at 1%); the evaluation script is released with the harness.

[2]https://huggingface.co/protectai/deberta-v3-base-prompt-injection-v2

VI. DISCUSSION

Forgotten knob or architectural limit? Neither. A forgotten knob would be closed by sending the whole conversation; it is not, in aggregate, for any of the seven guardrails or at any reconstructed strength (Section V-C), and the model-induced term is 0.36–1.00. A classifier that ignored history would score each decision the same with or without context; instead every commercial product that detects anything detects more per decision, and Model Armor's conversation-level FPR halves with AUROC unchanged. The default's edge is sampling, not perception (Section V-B), so running both configurations recovers almost nothing (Bedrock $+0.03$, Model Armor $+0.01$) at double the cost. A stateful wrapper that accumulates the per-turn scores buys detection under a fixed benign budget where full context buys none (Azure Text, Bedrock, Qwen3Guard). Proposition 2 bounds what any wider window can recover once the payload sits inside enough benign text.

Per-detector blind spots. The black-box rate hides that particular harm families slip past every internal filter of a product at once. Topic drift is the worst case throughout—Model Armor's aggregate detection of drift falls to 0.26 under full, and Azure Text catches 0.02—and 38% of drift conversations that explicitly solicit chemical or biological harm pass all fourteen Bedrock detectors. Buyers cannot see this from a pass/fail API, and it is exactly what a security review needs.

A cheap methodology. Because detection-only responses grade each detector, one pass recovers the full strength curve; the whole study cost a few hundred dollars. Any operator can run the released harness against their own tenant and configuration—the measurement that matters.

Threats to validity. Services update without notice; we pin guardrail and template versions and date-stamp every call. Public corpora may appear in vendor training data, which would inflate rather than depress detection. Benign and attack conversations have matching mean length, and re-weighting benign FPR to the attack length distribution changes it by at most 0.004. Bedrock's denied topics are ours, not the vendor's, so content-only figures are given alongside. Prompt Shields exposes only a binary verdict, so its curve cannot be reconstructed.

VII. RELATED WORK

Multi-turn attacks. Gradual escalation [1], human multi-turn jailbreaks [2], actor-driven drift [3], and automated multi-agent red teaming [4] show conversation-level attacks defeat

model-level defences; recent analysis cautions that multi-turn gains partly reflect resampling [5]. These target models; we use them as instruments against deployed guardrails.

Stateful defences. A growing line adds memory to the guardrail: recurrent intent-drift monitoring [7], session-level risk accumulation [8], cumulative monitors for decomposition [9], per-user buffers [10], and training on compressed dialogues [11]. These propose new detectors; none measures deployed commercial services across inspection configurations or at the per-detector level. Limits across unlinkable identities are established in [12].

Guardrail evaluation. A systematisation organises guardrails by safety granularity but evaluates open detectors, not deployed services [17]. Commercial APIs have been scored on fixed inputs: evasion by character injection [13], cross-product comparison [14], [25], adaptive attacks [15], and agent tool-calling trajectories [26]; window mismatch has been exploited within a single prompt [16]. An early study toggled conversation history for an open framework (NeMo Guardrails) and saw detection worsen [28], and long-context dilution of open guardrails has been measured mechanistically [27]. None varies what a deployed commercial service is handed, reconstructs its strength curve, or reports its internal detectors.

VIII. Conclusion

Commercial guardrails are sold as security controls and deployed to score one message at a time. Showing them the conversation does not help, and not because they ignore it: every product that detects anything detects more per decision with context; the turn-by-turn default wins per conversation by taking many weakly correlated looks. On four deployed services, with per-detector attribution, that mechanism turns "guardrails miss multi-turn attacks" into an account of what a deployment change can recover, which internal filters carry detection, which harm families evade all of them, and why padding bounds any wider window. We release the harness so operators can measure the control they already trust.

Ethical Considerations

This study evaluates classifiers, not models: published, labelled corpora are replayed to guardrail services in our own tenants in detection-only mode; no model is prompted to produce harmful content and no new attack technique is introduced. Corpus licences are respected; the released artifact contains code, identifiers and aggregate results, not harmful text. Findings were shared with the affected vendors on 1 September 2026 as configuration guidance, not as classifier vulnerabilities.

References

[1] M. Russinovich, A. Salem, and R. Eldan, "Great, now write an article about that: The Crescendo multi-turn LLM jailbreak attack," in *Proc. USENIX Security Symp.*, 2025.

[2] N. Li *et al.*, "LLM defenses are not robust to multi-turn human jailbreaks yet," arXiv:2408.15221, 2024.

[3] Q. Ren *et al.*, "LLMs know their vulnerabilities: Uncover safety gaps through natural distribution shifts," in *Proc. Annu. Meeting Assoc. Computational Linguistics (ACL)*, 2025.

[4] S. Rahman *et al.*, "X-Teaming: Multi-turn jailbreaks and defenses with adaptive multi-agents," in *Proc. Conf. Language Modeling (COLM)*, 2025.

[5] X. Yang *et al.*, "Multi-turn jailbreaks are simpler than they seem," arXiv:2508.07646, 2025.

[6] E. Yu *et al.*, "CoSafe: Evaluating large language model safety in multi-turn dialogue coreference," in *Proc. Conf. Empirical Methods in Natural Language Processing (EMNLP)*, 2024.

[7] J. Albrethsen, Y. Datta, K. Kumar, and S. Rajasekar, "DeepContext: Stateful real-time detection of multi-turn adversarial intent drift in LLMs," arXiv:2602.16935, 2026.

[8] S. Mishra, D. Chukkapalli, and G. R. Naik, "Stateful guardrails for multi-turn LLM systems: A conversational risk accumulation framework," arXiv:2607.19361, 2026.

[9] C. Yueh-Han, N. Joshi, Y. Chen, M. Andriushchenko, R. Angell, and H. He, "Monitoring decomposition attacks in LLMs with lightweight sequential monitors," in *Proc. Int. Conf. Learning Representations (ICLR)*, 2026.

[10] D. Brown *et al.*, "Benchmarking misuse mitigation against covert adversaries," arXiv:2506.06414, 2025.

[11] H. Kim, "Defensive M2S: Training guardrail models on compressed multi-turn conversations," arXiv:2601.00454, 2026.

[12] B. Sun, Z. Zhao, X. Liu, Y. Cao, and C. Xiao, "Decomposition attacks across unlinkable identities: Limits of stateful defenses for LLM services," arXiv:2608.17445, 2026.

[13] W. Hackett, L. Birch, S. Trawicki, N. Suri, and P. Garraghan, "Bypassing LLM guardrails: An empirical analysis of evasion attacks against prompt injection and jailbreak detection systems," arXiv:2504.11168, 2025.

[14] D. Kumar *et al.*, "No free lunch with guardrails," arXiv:2504.00441, 2025.

[15] M. Nasr *et al.*, "The attacker moves second: Stronger adaptive attacks bypass defenses against LLM jailbreaks and prompt injections," in *Proc. USENIX Security Symp.*, 2026.

[16] Y. Zhou *et al.*, "Prompt overflow: What the guardrail inspects is not what the model infers," arXiv:2605.23196, 2026.

[17] X. Wang *et al.*, "SoK: Evaluating jailbreak guardrails for large language models," in *Proc. IEEE Symp. Security and Privacy (S&P)*, 2026.

[18] Amazon Web Services, "ApplyGuardrail API reference," AWS documentation, accessed 2026.

[19] Microsoft, "Prompt Shields and Analyze Text, Azure AI Content Safety REST API, v2024-09-01," accessed 2026.

[20] Google Cloud, "Model Armor overview and sanitize APIs," accessed 2026.

[21] W. Zhao *et al.*, "WildChat: 1M ChatGPT interaction logs in the wild," in *Proc. Int. Conf. Learning Representations (ICLR)*, 2024.

[22] G. Bai *et al.*, "MT-Bench-101: A fine-grained benchmark for evaluating large language models in multi-turn dialogues," in *Proc. Annu. Meeting Assoc. Computational Linguistics (ACL)*, 2024.

[23] H. Zhao *et al.*, "Qwen3Guard technical report," arXiv:2510.14276, 2025.

[24] S. Chennabasappa *et al.*, "LlamaFirewall: An open source guardrail system for building secure AI agents," arXiv:2505.03574, 2025.

[25] M. Kang *et al.*, "Poly-Guard: Massive multi-domain safety policy-grounded guardrail dataset," in *Proc. NeurIPS Datasets and Benchmarks Track*, 2025.

[26] Y.-S. Chen, S.-Y. Huang, C.-L. Yang, and Y.-N. Chen, "TraceSafe: A systematic assessment of LLM guardrails on multi-step tool-calling trajectories," in *Proc. Conf. Language Modeling (COLM)*, 2026.

[27] Z. Chen, X. Wu, and S. Hu, "LongGuard: Mechanistic analysis and training-free mitigation of long-context failure in safety guardrails," arXiv:2608.27580, 2026.

[28] T. Gibbs *et al.*, "Emerging vulnerabilities in frontier models: Multi-turn jailbreak attacks," arXiv:2409.00137, 2024.