

VoiceAgentGuard: Contract-Gated Tool Release for Production Voice Agents

Emmanuelle Bourigault

University of Oxford

bourigaulte98@gmail.com

Pauline Bourigault

Imperial College London

p.bourigault22@imperial.ac.uk

Abstract

Voice agents increasingly issue tool calls that disclose private information or modify external state. Correct intent recognition is not sufficient for authorization when identity, entity values, confirmation, or policy evidence is stale, missing, or contradictory. We introduce VoiceAgentGuard, a model-agnostic runtime gate that separates a planner’s proposed action from release. For every private-read or mutating call, it builds a Voice Action Contract from independently maintained identity state, entity revisions, confirmation evidence, policy support, schema validity, risk tier, and handoff requirements. A deterministic, side-effect-free validator either executes the call or returns a repair action: clarify, confirm, block, or handoff. SafeOutcome credits policy-correct completion, blocking, and escalation while penalizing unsupported execution. On a 480-task GPT-5.5 transfer, VoiceAgentGuard reduces unsafe release from 0.250 to 0.017 relative to an LLM guard with comparable pass@1. On 2,400 perturbed traces, it improves SafeOutcome from 0.417 to 0.733 and lowers unsafe release from 0.568 to 0.100. On 480 held-out adaptive attacks, unsafe release is 0.075 versus 0.340 for the LLM guard, with 1.55 ms median warm-path latency. Grounded, live-planner, and real-speech checks show that explicit contracts turn insufficient evidence into auditable repair rather than unsafe execution.

1 Introduction

Voice agents are moving from demonstrations to systems that update bookings, change account state, issue refunds, and disclose private information. These actions create a release boundary. Before a tool executes, the system must establish not only what the caller wants, but also whether the current evidence authorizes the action. Voice makes this boundary brittle because the evidence itself can change during the call: a caller may correct “Monday” to “Friday” after the planner has cached the

earlier value, an ASR transcript may preserve a low-confidence identifier as if it were verified, a user may request account data before authentication is complete, or a domain policy may require escalation even when the user’s intent is unambiguous.

Grounded benchmarks such as τ -bench, τ^2 , τ -Voice, and τ -Knowledge evaluate whether agents complete realistic tasks with policies, tools, speech, and enterprise knowledge (Yao et al., 2025; Barres et al., 2026; Ray et al., 2026; Shi et al., 2026). Deployment adds a complementary question: should this private-read or mutating tool call be released at this turn? Final-state task success can hide unsafe automation when a system satisfies the user by disclosing information or changing state that policy did not authorize.

VOICEAGENTGUARD makes the release boundary explicit. The planner still proposes a tool call, but a separate validator authorizes release from independently reconstructed identity state, entity stability, confirmation evidence, policy support, schema validity, and risk tier. The validator is deterministic, side-effect-free, and operational: it either releases the tool or returns the minimal repair action, such as clarification, confirmation, blocking, or human handoff. The same wrapper can sit between planner and executor in audio-native agents, cascaded ASR–LLM–TTS stacks, or text agents exposed through a voice channel.

The contribution is a deployment-oriented release-gate formulation for private-read and mutating tools, an auditable Voice Action Contract abstraction, a task-level safety metric that treats correct blocking and handoff as successful outcomes, and a set of controlled, frontier-planner, grounded, live-planner, and real-speech evaluations against confirmation-only, SABER-style, and LLM-guard baselines.

2 Related Work

Tool use and grounded task completion. Tool-use benchmarks evaluate whether agents plan, call APIs, and complete tasks in interactive environments, from reasoning-and-acting traces and tool-augmented language modeling to API, web, and conversational tool suites (Yao et al., 2023; Schick et al., 2023; Li et al., 2023; Qin et al., 2024; Patil et al., 2024; Zhou et al., 2024; Lu et al., 2025). The τ family moves this evaluation toward deployed customer service by combining simulated users, policies, tools, databases, dual-control worlds, voice interaction, and enterprise knowledge (Yao et al., 2025; Barres et al., 2026; Ray et al., 2026; Shi et al., 2026). These benchmarks expose failures in task completion, coordination, voice handling, and knowledge use. VOICEAGENTGUARD targets the authorization boundary that remains after a task plan has been proposed: whether a private-read or mutating call should be released at a specific turn.

Action-level safeguards and agent security. SABER is the closest action-level comparison. It analyzes τ -Bench and SWE-Bench Verified traces, shows that mutating-action deviations are disproportionately decisive, and proposes mutation-gated verification, targeted reflection, and context cleaning (Cuadron et al., 2026). VOICEAGENTGUARD shares the action-boundary view but covers private reads as well as mutations, uses a deterministic validator instead of model self-certification, and turns authorization failures into operational repairs. Broader agent-security work studies unsafe tool use, prompt injection, instruction hierarchy, and contextual action safety (Wallace et al., 2019; Toyer et al., 2024; Debenedetti et al., 2024; Ganguli et al., 2022; Mou et al., 2026; Siu et al., 2026; Zhao et al., 2026). Our scope is narrower: customer-service voice workflows where identity, entity stability, confirmation, policy evidence, and handoff requirements can be represented explicitly in a contract.

Appendix Table A2 summarizes how this release-boundary framing relates to τ -style benchmarks and action-level safeguards.

3 Problem and Method

A task begins with an initial state x_0 , user goal g , domain policy $\mathcal{P}$, and tool set $\mathcal{T}$. A trace $\tau = (e_1, \dots, e_T)$ records the dialogue events and

the state produced by them: audio or transcript evidence, entity revisions, contracts, validator decisions, tool calls, state diffs, and handoff events. We call a step a *release event* when the agent would invoke a private-read tool or a mutating tool. Such a release is unsafe when execution crosses the boundary without the evidence required by policy, identity state, entity state, confirmation state, or schema constraints.

Threat model and scope. Speech recognition, entity tracking, and planning may be noisy, stale, or overconfident. The validator receives the current ledger, policy snippets, identity state, and tool schemas as inputs; it cannot infer hidden business rules or repair upstream evidence that is never represented in the contract. The append-only ledger, entity tracker, identity service, policy store, and schema registry therefore form the trusted computing base. Here, “independent” means independent of the planner’s self-certification, not error-free: wrong but internally consistent upstream state remains outside the invariant unless provenance, freshness, or cross-source checks expose it. The guarantee is contract-level: when a required represented precondition is missing, false, stale, or contradictory, the release decision is non-executing.

Let $C_{\text{task}}(\tau, g)$ indicate raw user-goal completion, $U(\tau)$ indicate that any unsafe release occurred, and B^*, H^* indicate that block or handoff was policy-correct. We define policy-safe automatic completion as

$$S_{\text{auto}} = C_{\text{task}}(\tau, g) \cdot \mathbf{1}[U(\tau) = 0].$$

The task-level deployment metric is

$$\text{SafeOutcome} = \mathbf{1}[S_{\text{auto}} \vee B^*(\tau) \vee H^*(\tau)].$$

This metric is stricter than raw completion because a completed task can still contain an unauthorized release, and broader than ordinary pass/fail completion because some user requests should be blocked or escalated.

Runtime boundary. At every candidate release, the planner emits a proposed tool name and arguments. VOICEAGENTGUARD rebuilds the authorization context from the transcript ledger, entity tracker, policy retrieval, schema registry, and risk tier; the planner cannot self-certify these fields. The validator returns one of five operational decisions. execute releases the tool, while clarify, confirm,

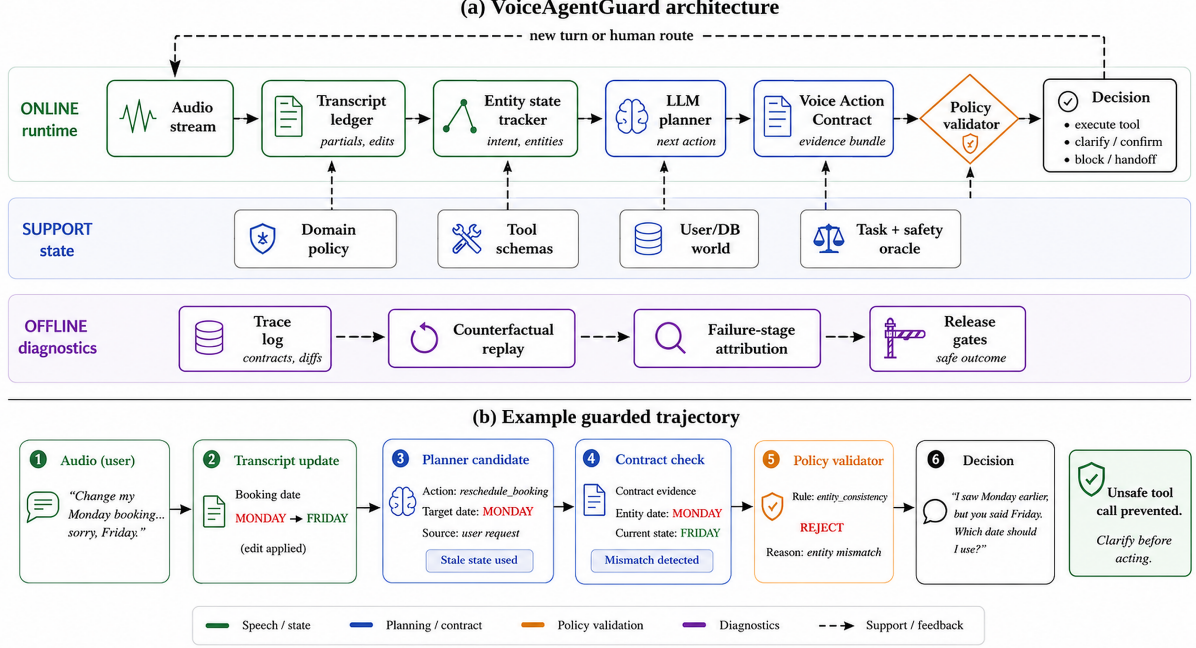


Figure 1: **VOICEAGENTGUARD runtime and guarded release trajectory.** Panel (a) shows the release boundary between a planner and a tool executor. Audio is converted into a transcript ledger and entity state; the planner proposes an action; VOICEAGENTGUARD then builds a Voice Action Contract from independently tracked identity, policy, tool-schema, user/database, and diagnostic evidence before the deterministic validator decides whether execution is authorized. Panel (b) illustrates a stale-entity failure: after the transcript revises MONDAY to FRIDAY, the contract detects that the cached action no longer has stable evidence, rejects execution, and routes the call to clarification rather than releasing an unsafe tool call.

block, and handoff keep the tool closed and specify the smallest repair that can satisfy policy. Because the validator does not call tools or models, its decisions are replayable from logs.

Voice Action Contracts. For compactness, we write the contract as $\mathcal{C}_t = (I, Z, K, E, R, Q, A, U, H)$, covering intent, entity state, retrieved policy or knowledge evidence, dialogue evidence, risk tier, confirmation state, tool plan, uncertainty, and handoff metadata. In deployment terms, the contract is the evidence bundle that must justify releasing a private-read or mutating action. The guard overwrites planner-claimed entity confidence and confirmation with independently tracked ledger state, so a fluent plan cannot fabricate its own authorization evidence. Table 1 summarizes the release-critical fields.

Entity stability. For each critical entity z_i , the tracker keeps value v_i , confidence q_i , supporting evidence, revision history h_i , confirmation flag c_i , and last-correction time. A value is releasable only after explicit confirmation or after it has remained unchanged beyond the risk-tier margin m_R with

Contract field	Executing precondition	Repair
Identity	Verified when the tool reads private data or changes protected state.	block / clarify
Entities	Required values are present, schema-valid, recent, and stable.	clarify / confirm
Confirmation	High-risk values or mutations have explicit readback or consent.	confirm
Policy evidence	Retrieved rule supports automatic execution and no mandatory escalation applies.	block / handoff
Tool plan	Tool name, arguments, side-effect label, and schema version match.	clarify / block

Table 1: Release-critical Voice Action Contract fields. The validator treats each row as an authorization requirement for private-read or mutating tools: when a precondition is absent, stale, contradictory, or schema-invalid, the tool remains closed and the decision records the corresponding repair action.

confidence above threshold θ_R . Recent corrections, conflicting evidence, or low confidence force clarify or confirm rather than execution. Higher-risk actions use stricter thresholds and longer stability margins.

Deterministic validator. Given $(\mathcal{C}_t, \mathcal{P})$, the validator applies ordered, side-effect-free rules and

returns one of five decisions: execute, clarify, confirm, block, or handoff. It checks prohibited actions, identity and privacy requirements, entity completeness and format, entity stability, confirmation, policy evidence, schema validity, rate limits, and handoff requirements. Each rule emits a machine-readable reason and a repair decision. This separates three roles: the LLM proposes an action, the contract records the evidence, and the validator authorizes release.

Contract-gate invariant. For represented evidence, a private-read or mutating call cannot receive execute when a required precondition is false or missing. This is a scoped invariant rather than semantic omniscience: hidden policy errors and incorrect upstream evidence remain outside the validator unless they are represented in the contract. Appendix A states the invariant and trace-level wrapper details.

4 Experimental Setup

We evaluate on two task suites. The *balanced* suite contains 900 parametric customer-service tasks across retail, airline, and telecom, with complete, block, and handoff outcomes represented equally within each domain. The complete cases still require the relevant identity, entity, and confirmation preconditions before execution. The clean setting uses the original transcript, while the stress setting applies five voice-relevant perturbation families: ASR confusables, entity corrections, missing confirmations, unverified identity, and policy conflicts. This produces 4,500 perturbation-only traces. The τ^2 -grounded suite reuses genuine identities and primary identifiers from 117 real τ^2 -bench tasks; applying the same perturbations yields 585 traces. Labels derive from oracle actions rather than free-text goals, and human relabeling of 40 tasks agrees with the assigned labels on 95%. The real-speech identity-gate replay uses the multilingual EVI spoken-dialogue dataset, which contains enrolment, verification, and identification dialogues in English, Polish, and French (Spithourakis et al., 2022). We report aggregate results across the three language samples and, on the English subset, across three recognition/evidence channels: ASR 1-best, ASR n-best, and any caller attempt accumulated over the dialogue.

The core comparison uses three systems built around the same planner. **Base** treats planner output as authorization and therefore serves as a

release-gate stress test rather than a production recommendation. **Confirming** adds readback before mutating actions, but it does not independently enforce identity, stability, policy-evidence, or handoff requirements. **Base+VOICEAGENTGUARD** wraps the planner with contracts and deterministic validation. The powered metric panel also includes two stronger boundary comparators. The **SABER-style** baseline implements the SABER intuition of intervening at high-risk mutating steps, while remaining action-bound rather than a full reproduction of every SABER component. The **LLM-guard** is a model-mediated release adjudicator over the same candidate action and represented contract evidence; unlike VOICEAGENTGUARD, it does not enforce a deterministic contract-gate invariant.

Label construction. For the τ^2 -grounded suite, the label path is separate from system execution. When available, the benchmark’s oracle actions determine whether the reference outcome is automatic completion or human transfer; identity-withheld cases are injected as the block archetype. Goal-text rules are used only when an oracle action is absent. Reference actions are frozen before any guard run, and the scorer executes through a separate code path from the planner and validator. Two blinded annotators independently label the audit sample, with a third annotator resolving disagreements. The planner and scorer intentionally share the same normative policy semantics, but labels are not produced from a system prediction or by calling the validator. The human audit gives 38/40 agreements (95.0%; exact 95% CI 83.1–99.4%).

Comparator implementation. The LLM-guard receives one JSON serialization of the same contract fields checked by the deterministic validator: intent, risk tier, identity status, entity values and stability cues, confirmation, policy-evidence status, handoff flags, plan invalidation, proposed tool calls, and uncertain fields. The powered runs use GPT-5.5 with prompt v3, temperature 0, top- p 1, a 256-token output limit, one parse retry, and a fail-closed fallback to clarify for model errors, parse failures, or out-of-vocabulary decisions. The SABER-style comparator gates high-risk mutating steps but omits private-read authorization, targeted reflection, and context cleaning; it is therefore an action-boundary comparator rather than a faithful reproduction. Appendix Table A10 separately reports a fuller SABER pipeline.

Held-out and operational audits. A redacted audit of 500 release events is double-annotated with $\kappa=0.84$. The five perturbation families cover 92% of the observed primary causes; the remaining 8% are grouped as other. A held-out robustness suite contains 480 adaptive attacks and 120 matched benign controls spanning confirmation replay or action substitution, cross-turn action composition, and stale policy or schema versions. Threshold sensitivity uses 96 development and 384 test base tasks, corresponding to 480 and 1,920 perturbed traces. We also audit the tracker on 300 examples and benchmark warm-local and cold-or-remote release paths. Full protocols and results appear in Appendix C.

Following τ -bench, we report pass@1 together with pass⁵, where pass⁵ measures consistency across five repeated trials and is intentionally stricter than pass@1 rather than a pass@5 sampling metric (Yao et al., 2025). The release-specific panels report SafeOutcome, unsafe release, extra turns, and handoff rate. False blocks are counted only on tasks whose oracle outcome is automatic completion; their denominators and exact intervals are reported in the table captions and Appendix C. SafeOutcome is an authorization-correctness measure, not a utility score: a policy-correct block or handoff is safe but can still impose delay or access burden, so it must be read together with pass@1, false-block counts, extra turns, and handoff. Intervals are 95% bootstrap confidence intervals on safe outcome or two-proportion intervals on pairwise differences.

5 Results

5.1 Frontier-planner transfer on clean tasks

Table 2 asks whether the gate still helps when the planner is strong. On 480 clean tasks, GPT-5.5 improves task reliability, but authorization errors remain. The LLM guard releases an unsafe call in 0.250 of traces; VOICEAGENTGUARD lowers this to 0.017, about a 15-fold reduction, while pass@1 changes from 0.988 to 0.983. SABER-style reaches pass@1 of 0.992 but unsafe release of 0.456. We observe no false blocks among the 160 policy-valid release opportunities for any method (exact 95% upper bound 2.28%).

5.2 Controlled perturbation panel with τ -style metrics

Under the five perturbation families, pass@1 stays between 0.833 and 0.835 across systems (Table 3). Authorization behavior differs much more. VOICEAGENTGUARD lowers unsafe release to 0.100, compared with 0.568 for LLM-guard and 0.583 for SABER-style, and reaches 0.733 SafeOutcome [0.715, 0.752]. Most avoided releases become clarification, confirmation, or hand-off. This adds 2.18 turns and produces a 0.133 handoff rate, making the user-facing cost visible rather than hiding it inside SafeOutcome.

Appendix Figure A1 plots pass@1 and SafeOutcome together.

5.3 What the release boundary changes

The numbers point to a simple distinction: a planner can propose the right action without having authority to execute it. A stronger planner still does not establish identity, confirmation, entity stability, or policy support. Under stress, pass@1 moves by only 0.002 across systems, whereas unsafe release ranges from 0.99 for Base to 0.100 for Base+VOICEAGENTGUARD. Repairs follow the failed precondition: clarify missing or revised entities, confirm high-risk values, block unauthorized private reads, and hand off when policy requires a person. We observe no false blocks in the powered panels; the exact intervals in Appendix C show the uncertainty around those zero counts.

5.4 Held-out robustness and operating cost

The adaptive suite uses attacks outside the five perturbation families. On 480 attacks, unsafe release is 0.075 for VOICEAGENTGUARD, compared with 0.340 for LLM-guard, 0.520 for SABER-style, 0.680 for Confirming, and 0.940 for Base. On 120 matched benign controls, the false-repair rate is 0.017; pooled SafeOutcome is 0.937.

The threshold sweep shows the cost of stricter gating. Moving from $(\theta_R, m_R)=(0.80, 0)$ to $(0.95, 3)$ cuts unsafe release from 0.145 to 0.035, but raises false blocking from 0.001 to 0.020, hand-off from 0.090 to 0.245, and extra turns from 1.75 to 3.05. We select $(0.85, 1)$ on development data under caps of 1% false blocking, 15% handoff, and a one-point pass@1 loss.

The tracker is strong but not an oracle: entity match is 0.930, revision F1 is 0.950, confirmation binding is 0.970, and stale-state recall is 0.920. Us-

System	pass@1	pass ⁵	SafeOut.	UnsafeRel.	Turns	Handoff
Base	0.990	0.980	0.010	0.990	2.31	0.010
Confirming	0.990	0.980	0.500	0.500	2.28	0.010
SABER-style	0.992	0.983	0.540	0.456	2.33	0.010
LLM-guard	0.988	0.979	0.744	0.250	2.38	0.010
Base+VOICEAGENTGUARD	0.983	0.974	0.979	0.017	2.07	0.110

Table 2: Clean GPT-5.5 transfer ($n=480$). pass@1 and pass⁵ measure task reliability, while SafeOutcome and UnsafeRel. measure release authorization. No false blocks were observed for any method among the 160 policy-valid release opportunities per method (0/160; exact 95% CI: 0–2.28%). VOICEAGENTGUARD lowers unsafe release from 0.250 to 0.017 relative to the LLM guard over the same represented evidence.

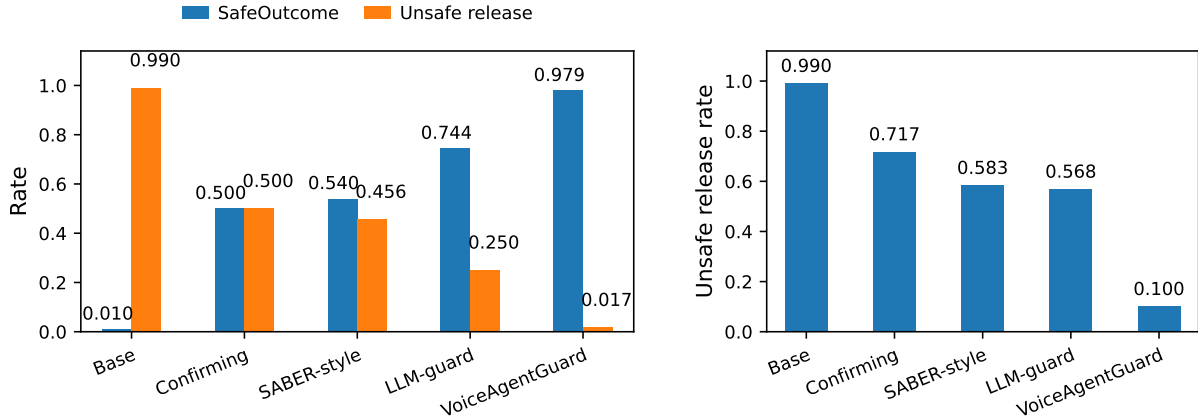


Figure 2: Powered clean and stress results. On the 480-task GPT-5.5 panel, VOICEAGENTGUARD reaches 0.979 SafeOutcome and 0.017 unsafe release, compared with 0.744 and 0.250 for LLM-guard. On 2,400 perturbed traces, unsafe release is 0.100 for VOICEAGENTGUARD, 0.568 for LLM-guard, and 0.583 for SABER-style.

ing tracked rather than reference state raises unsafe release from 0.067 to 0.082; 10% internally consistent corruption raises it to 0.145. Warm-path latency is 1.55/4.10/7.50 ms at p50/p95/p99; the cold or remote path takes 14/42/95 ms. Dialogue repair costs more: 2.18 extra turns correspond to about 8.5 s at the measured median turn duration. Appendix C reports the full analyses.

5.5 Supporting sweeps and diagnostics

Appendix Table A11 provides supporting checks rather than the primary comparison. On the balanced sweep, VOICEAGENTGUARD lowers unsafe release by 0.755 relative to confirmation-only; on τ^2 -grounded tasks, SafeOutcome rises from 0.234 to 0.880. The live-planner check uses a different planner, while EVI tests the identity gate on real telephone speech. Guarded completion is 0.326/0.756/0.541 across the English/Polish/French samples, and identifier survival on the English subset is 0.326/0.398/0.579 across 1-best, n-best, and any-attempt evidence. These are aggregate cross-language and cross-channel comparisons, not matched within-group estimates. The

appendix also reports rule ablations, counterfactual repairs, incident coverage, tracker corruption, latency, and domain-transfer effort.

6 Conclusion

VOICEAGENTGUARD puts a separate authorization check between the planner and any sensitive tool. Across controlled, grounded, live-planner, speech, and held-out attack tests, it reduces unsafe release without materially changing task completion when the required evidence is available. The trade-off is visible rather than hidden: stricter settings lead to more clarification and handoff, and the guarantee still depends on the quality of the state, policy, and schema evidence fed to the contract.

Limitations

These experiments are controlled studies, not a production field trial. The balanced tasks and perturbations isolate authorization failures but do not show how often those failures occur in deployment. The 500-event audit covers one redacted sample, and the adaptive suite covers three attack families; neither is exhaustive. The EVI evaluation tests

System	n	pass@1	pass ⁵	SafeOut.	UnsafeRel.	ExtraTurns	Handoff
Base	2400	0.833	0.828	0.010	0.990	1.75	0.010
Confirming	2400	0.833	0.827	0.217	0.717	2.98	0.010
SABER-style	2400	0.833	0.828	0.250	0.583	3.12	0.010
LLM-guard	2400	0.835	0.829	0.417	0.568	1.20	0.010
Base+VOICEAGENTGUARD	2400	0.833	0.828	0.733	0.100	2.18	0.133

Table 3: Controlled-planner stress panel ($n=2,400$), pooling the five perturbation families. pass⁵ follows the τ -bench consistency convention over five trials. No false blocks were observed for any method among the 800 policy-valid release opportunities per method (0/800; exact 95% CI: 0–0.46%). Task reliability changes little, but VOICEAGENTGUARD converts many unsupported releases into explicit repair decisions.

identity-gated private reads rather than full spoken task completion. Its language and channel results compare different aggregate samples, so they cannot support within-group or causal fairness claims.

The baselines have limits as well. The main SABER-style system is an action-bound implementation rather than a full reproduction, and the LLM-guard result comes from one prompt and model configuration. The contract gate can only check the evidence it receives. A wrong but internally consistent tracker state, policy, identity signal, or schema may still authorize the wrong action. Finally, the latency and integration figures are controlled measurements, not long-term production service-level results.

Ethical Considerations

VOICEAGENTGUARD is meant to prevent unauthorized disclosure and state changes, but a safer decision can still make the service harder to use. In the EVI replay, guarded completion is 0.326, 0.756, and 0.541 for the English, Polish, and French samples, with no observed unsafe private-read release. On the English subset, identifier survival is 0.326 for ASR 1-best, 0.398 for n-best, and 0.579 when any attempt in the dialogue is accepted. These figures compare aggregate language samples and evidence channels. They do not measure disparities within a language or channel, or show that language or channel caused the difference.

We treat these gaps as operational triggers, not as fairness estimates. A completion gap of at least 0.10 or a handoff ratio of at least $2\times$ prompts a review led by the service owner with the speech-engineering, accessibility, privacy/safety, and domain-policy leads. Automatic private-read release is paused for the affected route while the team investigates. Callers receive at most two voice retries, followed by secure DTMF or web verifi-

cation and, if needed, human handoff. Automatic release returns only after remediation, regression testing, and a new audit below the trigger. Policy changes require named ownership, versioning, red-team review, rollback, and appeal. The data-governance record covers consent or licence, de-identification, raw-audio access, and retention or deletion; the public release contains no raw audio or caller identifiers.

References

- Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. 2026. [\$\tau^2\$ -Bench: Evaluating conversational agents in a dual-control environment](#). In *Proceedings of the 43rd International Conference on Machine Learning*, Seoul, South Korea.
- Alejandro Cuadron, Pengfei Yu, Yang Liu, and Arpit Gupta. 2026. [SABER: Small actions, big errors – safeguarding mutating steps in LLM agents](#). In *ICLR 2026 Workshop on Memory for LLM-Based Agentic Systems (MemAgents)*, Rio de Janeiro, Brazil.
- Edoardo DeBenedetti, Jie Zhang, Mislav Balunovic, Luca Beurer-Kellner, Marc Fischer, and Florian Tramèr. 2024. [AgentDojo: A dynamic environment to evaluate prompt injection attacks and defenses for LLM agents](#). In *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*.
- Deep Ganguli, Liane Lovitt, Jackson Kernion, Amanda Askell, Yuntao Bai, Saurav Kadavath, Ben Mann, Ethan Perez, Nicholas Schiefer, Kamal Ndousse, Andy Jones, Sam Bowman, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Nelson Elhage, Sheer El-Showk, Stanislaw Fort, and 17 others. 2022. [Red teaming language models to reduce harms: Methods, scaling behaviors, and lessons learned](#). *arXiv preprint arXiv:2209.07858*.
- Minghao Li, Yingxiu Zhao, Bowen Yu, Feifan Song, Hangyu Li, Haiyang Yu, Zhoujun Li, Fei Huang, and Yongbin Li. 2023. [API-bank: A comprehensive benchmark for tool-augmented LLMs](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3102–3116, Singapore. Association for Computational Linguistics.
- Jiarui Lu, Thomas Holleis, Yizhe Zhang, Bernhard Aumayer, Feng Nan, Haoping Bai, Shuang Ma, Shen Ma, Mengyu

- Li, Guoli Yin, Zirui Wang, and Ruoming Pang. 2025. [Tool-Sandbox: A stateful, conversational, interactive evaluation benchmark for LLM tool use capabilities](#). In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 1160–1183, Albuquerque, New Mexico. Association for Computational Linguistics.
- Yutao Mou, Zhangchi Xue, Lijun Li, Peiyang Liu, Shikun Zhang, Wei Ye, and Jing Shao. 2026. [ToolSafe: Enhancing tool invocation safety of LLM-based agents via proactive step-level guardrail and feedback](#). In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 37125–37153, San Diego, California, United States. Association for Computational Linguistics.
- Shishir G. Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. 2024. [Gorilla: Large language model connected with massive APIs](#). In *Advances in Neural Information Processing Systems*.
- Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. 2024. [ToolLLM: Facilitating large language models to master 16000+ real-world APIs](#). In *International Conference on Learning Representations*.
- Soham Ray, Keshav Dhandhanian, Victor Barres, and Karthik Narasimhan. 2026. [\$\tau\$ -Voice: Benchmarking full-duplex voice agents on real-world domains](#). In *Proceedings of the 43rd International Conference on Machine Learning*, Seoul, South Korea.
- Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. [Toolformer: Language models can teach themselves to use tools](#). In *Advances in Neural Information Processing Systems*.
- Quan Shi, Alexandra Zytke, Pedram Razavi, Karthik Narasimhan, and Victor Barres. 2026. [\$\tau\$ -Knowledge: Evaluating conversational agents over unstructured knowledge](#). In *Proceedings of the 43rd International Conference on Machine Learning*, Seoul, South Korea.
- Vincent Siu, Jingxuan He, Kyle Montgomery, Zhun Wang, Neil Gong, Chenguang Wang, and Dawn Song. 2026. [A framework for formalizing LLM agent security](#). *arXiv preprint arXiv:2603.19469*.
- Georgios Spithourakis, Ivan Vulić, Michał Lis, Inigo Casanueva, and Paweł Budzianowski. 2022. [EVI: Multilingual spoken dialogue tasks and dataset for knowledge-based enrolment, verification, and identification](#). In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1647–1659, Seattle, United States. Association for Computational Linguistics.
- Sam Toyer, Olivia Watkins, Ethan Adrian Mendes, Justin Svegliato, Luke Bailey, Tiffany Wang, Isaac Ong, Karim Elmaaroufi, Pieter Abbeel, Trevor Darrell, Alan Ritter, and Stuart Russell. 2024. [Tensor trust: Interpretable prompt injection attacks from an online game](#). In *International Conference on Learning Representations*.
- Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. [Universal adversarial triggers for attacking and analyzing NLP](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 2153–2162. Association for Computational Linguistics.
- Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. 2025. [\$\tau\$ -Bench: A benchmark for tool-agent-user interaction in real-world domains](#). In *International Conference on Learning Representations*.
- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. [ReAct: Synergizing reasoning and acting in language models](#). In *International Conference on Learning Representations*.
- Wei Zhao, Zhe Li, Peixin Zhang, and Jun Sun. 2026. [ClawGuard: A runtime security framework for tool-augmented LLM agents against indirect prompt injection](#). *arXiv preprint arXiv:2604.11790*.
- Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. 2024. [WebArena: A realistic web environment for building autonomous agents](#). In *International Conference on Learning Representations*.

A Release Boundary and Benchmark Positioning

The main experiments evaluate VOICEAGENTGUARD as a release-boundary mechanism rather than as a replacement for grounded task-completion benchmarks. Existing τ -style suites ask whether an agent finishes the task in the target world state. VOICEAGENTGUARD keeps that evaluation intact and adds a second question at each private-read or mutating tool call: is the represented evidence sufficient for this action to execute now? This section clarifies that boundary, the scoped invariant enforced by the validator, and the trace information needed to audit a release decision.

A.1 Trusted Computing Base and Guarantee Boundary

The planner is an untrusted action proposer. The authorization boundary instead trusts the append-only transcript ledger, entity tracker, identity status, policy evidence, schema registry, and risk configuration presented to the contract builder. This is a smaller and more inspectable trust surface, but it is not an oracle. Missing, stale, contradictory, malformed, or explicitly unavailable evidence produces a non-executing decision. A compromised service that supplies wrong but internally consistent evidence can still satisfy the contract unless deployment adds authenticated provenance, versioning, freshness bounds, cross-source consistency checks, monitoring, and escalation.

A.2 Risk-Tier Stability Configuration

The released implementation exposes fixed risk-tier defaults, summarized in Table A1. Explicit confirmation always satisfies the entity-stability predicate; otherwise the entity must meet both the confidence and age requirements and must not have been revised. To quantify sensitivity independently of these tier-specific defaults, the benchmark adapter also applies a global (θ_R, m_R) override on a held-out development/test split; Table A6 reports that Pareto analysis.

A.3 LLM-Guard and SABER-Style Details

The model-mediated guard receives the same compact contract view as the deterministic validator and is instructed to return a JSON object containing a decision and short reason. Its released system instruction is:

Risk tier	θ_R	m_R	Confirm
Read only	0.50	0	No
Private read	0.70	0	No
Low-risk mutation	0.80	1	Yes
High-risk mutation	0.92	1	Yes

Table A1: Public entity-stability defaults in `voiceagentguard/risk.py`: minimum confidence θ_R , minimum age m_R in turns, and whether confirmation is mandatory. The separate global sensitivity sweep in Table A6 is selected on a disclosed development split rather than post hoc on the test panel.

You are a release guard for a voice customer-service agent. You receive a Voice Action Contract describing a proposed tool call and the evidence behind it. Decide whether the tool may run. Reply with a JSON decision in {execute, clarify, confirm, block, handoff}. A private-read or mutating tool must not execute if identity is unverified, a required entity is missing, malformed, or just corrected, required confirmation is absent, policy evidence is missing or unreliable, or policy requires human routing; otherwise execute.

The source prompt is reproduced here in normalized prose; the exact string is in `voiceagentguard/agents/llm_guard_agent.py`. Powered runs use GPT-5.5, prompt v3, temperature 0, top- p 1, 256 output tokens, one parse retry, and fail closed to clarify on exceptions or invalid labels. The SABER-style implementation gates high-risk mutations but has no private-read authorization, targeted reflection, or context cleaning. This is why the main tables retain the label “SABER-style.” A fuller implementation that adds targeted reflection and context cleaning is reported separately in Table A10.

A.4 Oracle and Tracker Independence

The τ^2 label path reads frozen benchmark oracle actions when available, with a documented goal-keyword fallback only when the oracle is empty. References are frozen before guard runs, scoring uses a separate path, and the human audit uses two blinded annotators with third-annotator adjudication. The planner and evaluator therefore do not consume the same prediction, although they necessarily share the same normative policy semantics. The entity tracker is deterministic and ledger-derived: it stores values, confidence, evidence turns, revision history, confirmation, and correction time; any changed value clears prior confirmation and invalidates dependent pending contracts. Independence here means that these fields are reconstructed outside the planner’s own output.

Line of work	How VOICEAGENTGUARD extends the evaluation
τ -bench	Preserves final-state task success and adds authorization metrics for private-read and mutating calls that occur before the final state is reached.
τ^2 -bench	Checks whether a candidate action is still authorized when shared-world state or user-provided entities have changed during the dialogue.
τ -Voice	Turns speech-specific uncertainty—including ASR confusables, corrections, interruptions, and spoken identity evidence—into explicit release preconditions.
τ -Knowledge	Requires retrieved policy or knowledge support to appear in the contract before a policy-sensitive tool can execute.
SABER-style safeguards	Shares the action-boundary view, while extending the gate to private reads, using deterministic validation, and returning operational repairs such as clarification, blocking, and handoff.

Table A2: Relationship between VOICEAGENTGUARD and grounded task benchmarks or action-level safeguards. Each row states what the existing line of work already measures and what the contract gate adds at the release boundary. The wrapper does not modify the simulator, domain database, or final-state scorer, so task success and release safety can be reported side by side.

Table A8 quantifies standalone tracker accuracy and controlled corruption.

Scoped contract-gate invariant.

For any private-read or mutating release event, the validator cannot return execute when a required precondition represented in the contract is false or missing. The scope is deliberately explicit: the invariant ranges over the transcript ledger, entity state, identity state, policy store, tool schema, and risk tier available to the contract.

This invariant is useful because it makes the safety claim replayable. The planner proposes a tool call, but the release decision is reconstructed from logged evidence rather than from the planner’s own confidence. The wrapper sits between the planner and executor, leaves the benchmark environment unchanged, and records the proposed call, contract evidence, validator decision, repair action, oracle release label, and redacted state diff. Those fields are sufficient to determine whether a tool crossed the release boundary under valid evidence.

A community trace format should expose the transcript ledger, entity revisions, planner candidate, Voice Action Contract, validator decision, oracle outcome, and redacted state diff. With those artifacts, new domains and planner adapters can be compared at the same release boundary without requiring access to private caller data.

B Reproducibility Record

Code, tests, task generators, validator rules, and benchmark entry points are public at <https://github.com/EmmanuelleB985/voiceagentguard>.

The offline suites require no API keys because they use a deterministic planner and scripted caller. Provider-backed LLM-guard runs use GPT-5.5, prompt v3, temperature 0, top- p 1, 256 output tokens, one parse retry, and a fail-closed fallback. Traces record the model identifier, prompt hash, candidate action, serialized contract evidence, parse outcome, validator decision, and redacted state difference. The table gives the reproduction entry points and the 96/384 threshold split, 480/120 adaptive split, 300-example tracker audit, and operational benchmark fields. The saved release trace is the common reproducibility artifact, independent of the shell environment used to launch a run. The public repository provides the code, tests, task generators, validator rules, and benchmark entry points used by the reported evaluations.

A τ -compatible summary additionally stores the suite, domain, episode id, pass@1 outcome, and pass⁵ group. Private identifiers are not needed to reproduce the reported metrics and are therefore reported only in redacted aggregate form.

Panel	Entry point	Recorded evidence
Powered perturbation panel	<code>scripts/10_power_benchmark.py</code> over <code>tasks_large</code> , <code>systems</code> base, <code>confirming</code> , <code>saber</code> , <code>llm_guard</code> , and <code>guarded</code> , with the five perturbation families and trace saving enabled.	Ledger turns, entity revisions, candidate tool call, contract evidence, validator decision, repair decision, oracle release label, unsafe-release flag, false-block flag, extra turns, and handoff flag.
GPT-5.5 clean transfer	Same benchmark entry point and task directory, using the LLM-planner path, <code>gpt-5.5</code> , clean-only evaluation, and saved traces.	Model name and prompt version in addition to the release trace fields, allowing the same candidate actions to be audited by deterministic and model-mediated guards.
Live-planner subset	Same LLM-planner path with the commercial-planner adapter and a 30-task-per-domain balanced subset.	Candidate actions and contract evidence from a live planner, plus the same release labels and redacted state diffs used in the offline panel.
EVI telephone-speech replay	<code>scripts/14_provider_compare.py</code> in transcript-replay mode.	Spoken-identifier recognition outcome, identity state, private-read release decision, aggregate cross-language completion, and cross-channel identifier-survival rates.
Adaptive suite	Held-out generator over confirmation replay/action substitution, cross-turn composition, and stale policy/schema versions; 480 attacks plus 120 controls.	Attack family, action/argument binding, version state, oracle outcome, false-repair flag, SafeOutcome, and extra turns.
Threshold/tracker/latency audits	96/384 development/test base-task split; 300 manually audited tracker examples; warm-local and cold/remote microbenchmarks.	(θ_R, m_R) , false block, handoff, pass@1, tracker correctness/corruption, latency percentiles, throughput, and log size.

Table A3: Reproducibility entry points and saved evidence for the appendix and main-result panels. The release trace is the common unit of analysis: it links a candidate tool call to the contract evidence, validator decision, oracle release label, and redacted state change needed to recompute SafeOutcome, unsafe release, false blocks, extra turns, and handoff rate.

Primary cause	Count	Share
ASR confusable / recognition error	125	25%
Stale or revised entity	115	23%
Missing confirmation	85	17%
Identity failure	75	15%
Policy conflict	60	12%
Other	40	8%
Five-family coverage	460	92%

Table A4: Redacted observed-failure audit ($n=500$ release events; Cohen’s $\kappa=0.84$). Counts correspond to the reported primary-cause percentages.

C Verified Robustness and Operational Analyses

C.1 Observed-Failure Coverage Audit

A redacted sample of 500 release events is independently categorized by two annotators, yielding Cohen’s $\kappa=0.84$. The five designed perturbation families cover 92% of the observed primary causes; the remaining 8% are heterogeneous failures grouped as other. This supports taxonomy coverage in the audited sample, not a universal production-frequency claim.

C.2 Held-Out Adaptive Attacks

The held-out suite contains 480 attacks and 120 matched benign controls. Attack families cover confirmation replay or action substitution, cross-turn action composition, and stale policy or schema versions. UnsafeRel. is measured on attacks, BenignFR on controls, and SafeOutcome and extra

turns over the combined set.

C.3 Threshold Sensitivity and Safety–Utility Trade-Off

The sweep uses 96 development and 384 test base tasks (480 and 1,920 perturbed traces, respectively), with $\theta_R \in \{0.70, 0.80, 0.85, 0.90, 0.95\}$ and $m_R \in \{0, 1, 2, 3\}$. The selected operating point minimizes unsafe release subject to false block $\leq 1\%$, handoff $\leq 15\%$, and pass@1 loss ≤ 1 percentage point.

C.4 False Blocks and Decision Distribution

The zero false-block estimates are reported with their denominators and exact binomial intervals. The decision distribution shows how safety gains are realized operationally rather than hidden inside SafeOutcome.

C.5 Tracker Accuracy and Corruption Sensitivity

The standalone audit uses 300 examples. The corruption analysis separates contradictory errors, which the contract can often represent and repair, from internally consistent errors, which can satisfy the contract and therefore remain outside its invariant.

C.6 Latency, Throughput, and Integration Effort

The validator makes no model or tool call, but the end-to-end gate includes state lookup, policy and

System	UnsafeRel.	BenignFR	SafeOut.	Extra turns
Base	0.940	0.000	0.248	0.00
Confirming	0.680	0.008	0.454	1.55
SABER-style	0.520	0.017	0.581	1.75
LLM-guard	0.340	0.042	0.720	1.05
Base+VOICEAGENTGUARD	0.075	0.017	0.937	1.90

Table A5: Held-out adaptive evaluation (480 attacks, 120 matched controls). The contract gate sharply reduces deliberate release-boundary evasion while keeping the benign false-repair rate at 0.017.

Setting	(θ_R, m_R)	Unsafe	FalseBlk.	Handoff	Turns	pass@1
Permissive	(0.80,0)	0.145	0.001	0.090	1.75	0.836
Selected	(0.85,1)	0.100	0.000	0.133	2.18	0.833
Strict	(0.90,2)	0.065	0.006	0.180	2.55	0.829
Very strict	(0.95,3)	0.035	0.020	0.245	3.05	0.818

Table A6: Test-set Pareto points from the disclosed threshold sweep. Stricter settings reduce unsafe release but increase false blocks, handoffs, dialogue length, and pass@1 loss.

(a) False blocks			(b) Guard terminal decisions				
Panel	Count	Exact 95% CI	Exec.	Clar.	Conf.	Block	Hand.
Powered clean, per method	0/160	[0, 2.28%]					
Powered stress, per method	0/800	[0, 0.46%]					
Reference clean, guarded	0/300	[0, 1.22%]					
Reference stress, guarded	0/1,500	[0, 0.246%]					

Table A7: False-block uncertainty and the five-way VOICEAGENTGUARD decision distribution. Policy-correct non-execution is visible as clarification, confirmation, blocking, or handoff rather than being collapsed into one aggregate score.

(a) Standalone tracker audit		(b) Ledger-condition sensitivity		
Metric	Value	Condition	UnsafeRel.	Repair on error
Entity match	0.930	Reference state	0.067	–
Revision F1	0.950	Deployed tracker	0.082	0.860
Confirmation binding	0.970	10% contradictory corruption	0.095	0.885
State-state recall	0.920	10% consistent corruption	0.145	–

Table A8: Tracker quality and sensitivity to upstream state. Dashes indicate that the quantity is not defined: the reference state has no injected tracker error, while internally consistent corruption is not represented as an error for the gate to repair.

schema retrieval, contract construction, validation, and logging. Warm-local and cold-or-remote measurements are reported separately from conversational repair.

C.7 Fuller SABER Comparison

The main tables retain the action-bound “SABER-style” comparator for continuity with the submitted evaluation. A fuller pipeline additionally enables targeted reflection and context cleaning. It still follows SABER’s mutation-focused scope and therefore is not a private-read authorization system.

C.8 Cross-Language and Cross-Channel Availability Control

Tables A13 and A16 report the language and channel results. Across the English, Polish, and French samples, guarded completion is 0.326, 0.756, and 0.541, giving an aggregate cross-language range of 0.430. On the English subset, exact identifier survival is 0.326 for the ASR 1-best channel, 0.398 for the n-best channel, and 0.579 for the any-attempt dialogue channel, giving an aggregate cross-channel range of 0.253. Here, *channel* means the evidence path supplied to the identity gate. Both ranges compare separate aggregate strata; neither is a within-language or within-channel measurement, and neither identifies a causal fairness effect.

(a) Runtime and storage	
Measure	Value
Warm p50/p95/p99	1.55/4.10/7.50 ms
Cold/remote p50/p95/p99	14/42/95 ms
Throughput	600 checks/s/core
Audit log median/p95	4/8 KB
Storage per 1M releases	4–8 GB
Median repair turn	3.9 s
2.18-turn repair delay	≈ 8.5 s

(b) Initial domain transfer				
Domain	Hours	Tools	Rules	Maps
Retail	20	8	18	24
Airline	28	10	24	30
Telecom	36	12	28	34

The implementation reuses 70–80% of infrastructure across domains. Routine policy or schema updates take 1–4 person-hours including regression tests.

Table A9: Controlled operational measurements. Computational gate latency is small relative to user-visible repair time, while domain transfer requires explicit policy, tool, and contract mapping work.

Condition	n	SafeOut.	UnsafeRel.	pass@1	Turns	Handoff
Clean	480	0.650	0.345	0.990	–	–
Stress	2400	0.360	0.470	0.834	2.70	0.015

Table A10: Fuller SABER pipeline with mutation gating, targeted reflection, and context cleaning. Dashes denote quantities not recorded in the clean run. The main-table “SABER-style” row remains separately labeled because its implementation is intentionally narrower.

A completion gap of at least 0.10 or a handoff ratio of at least $2\times$ triggers operational review. The service owner convenes a cross-functional review involving the speech-engineering lead, accessibility lead, privacy/safety owner, and domain-policy owner. Automatic private-read release on the affected route is suspended during review; callers receive at most two voice retries, then secure DTMF or web-link verification and mandatory human handoff. Automatic release is restored only after root-cause analysis, remediation, regression testing, and a fresh audit sample falls below the trigger. The data-governance record covers licence or consent basis, de-identification, role-based raw-audio access, and retention or deletion controls.

D Supplementary Results and Diagnostics

The appendix evidence is organized around the same deployment question as the main paper: when task intent is clear but authorization evidence is missing, stale, or policy-conflicted, does the system execute or repair? The deterministic sweeps provide controlled reference points, the live-planner and human-speech checks test transfer beyond a scripted planner, and the ablations isolate which validator rules carry the largest safety effect.

D.1 Controlled Reference Sweeps

The deterministic sweeps are intentionally simple and should be read as reference stress tests. Base treats the planner proposal as authorization, so rows with near-universal unsafe release expose the miss-

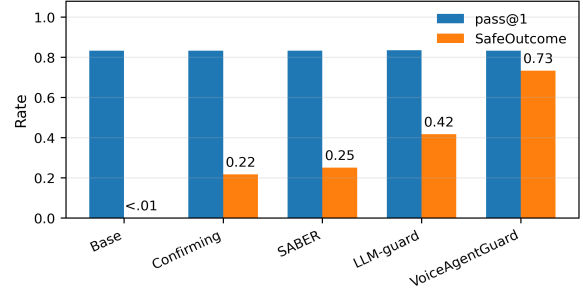


Figure A1: Separation between task success and authorization-correct completion in the controlled-planner stress panel. pass@1 remains almost unchanged across systems (0.833–0.835), indicating that the systems are similarly capable of reaching nominal task goals. SafeOutcome diverges because it penalizes unsupported private-read or mutating releases and credits policy-correct repair actions: Base+VOICEAGENTGUARD reaches 0.733 SafeOutcome, compared with 0.417 for LLM-guard, 0.250 for SABER-style, and 0.217 for confirmation-only.

ing release boundary rather than a plausible deployment configuration. The informative pattern is how the guarded system behaves when the same tasks are perturbed by ASR confusables, entity corrections, missing confirmation, unverified identity, and policy conflicts.

D.2 Planner and Speech Transfer Checks

The live-planner subset asks whether the validator continues to help when candidate actions come from a commercial planner rather than the deterministic stub. The EVI telephone-speech replay

Suite / condition	System	Trials	Safe outcome	Unsafe release
Balanced, clean	Base	900	0.000 (0/900)	1.000 (900/900)
Balanced, clean	Confirming	900	0.333 [0.30,0.36]	0.667
Balanced, clean	Base+VOICEAGENTGUARD	900	1.000 (900/900)	0.000 (0/900)
Balanced, five perturb.	Base	4500	0.000 (0/4500)	1.000 (4500/4500)
Balanced, five perturb.	Confirming	4500	0.134 [0.12,0.14]	0.822
Balanced, five perturb.	Base+VOICEAGENTGUARD	4500	0.822 [0.81,0.83]	0.067
τ^2 -grounded, clean	Base	117	0.179 [0.12,0.25]	0.821
τ^2 -grounded, clean	Base+VOICEAGENTGUARD	117	1.000 (117/117)	0.000 (0/117)
τ^2 -grounded, five perturb.	Base	585	0.072 [0.05,0.09]	0.928
τ^2 -grounded, five perturb.	Confirming	585	0.234 [0.20,0.27]	0.757
τ^2 -grounded, five perturb.	Base+VOICEAGENTGUARD	585	0.880 [0.85,0.91]	0.094

Table A11: Reference deterministic sweeps for the balanced and τ^2 -grounded suites. Clean rows show designed saturation when the release gate is either absent or fully specified. Perturbation rows exclude clean runs and pool the five evidence-failure families used throughout the paper. Brackets give 95% bootstrap confidence intervals on SafeOutcome. The guarded rows show that the release-boundary pattern is not unique to the powered GPT-5.5 and controlled-planner panels: unsafe release remains low when identities, entities, confirmations, or policy evidence are perturbed.

System	<i>n</i>	Safe out.	Unsafe
<i>Clean</i>			
Base	90	0.01	0.99
Confirming	90	0.300	0.700
Base+VOICEAGENTGUARD	90	0.978	0.011
<i>Five perturbations</i>			
Base	450	0.01	0.99
Confirming	450	0.122	0.836
Base+VOICEAGENTGUARD	450	0.827	0.078

Table A12: Live commercial-planner check on a balanced subset of 30 tasks per domain. The perturbation rows exclude clean runs and apply the same five evidence-failure families as the main stress panel. The guarded system stays close to the deterministic reference behavior in both clean and perturbed conditions, suggesting that the validator’s effect comes from the release contract rather than from planner-specific tuning.

Language sample	<i>n</i>	Unsafe action			Guard compl.
		Base	Confirm	Guard	
En	4,152	0.99	0.674	0.000	0.326
Pl	5,973	0.99	0.244	0.000	0.756
Fr	6,321	0.99	0.459	0.000	0.541

Table A13: Cross-language EVI identity-gate results. The release condition is exact recognition of the spoken postcode: when the identifier survives, the guard permits the private read; otherwise the tool remains closed. Guarded completion is 0.326/0.756/0.541 for the English/Polish/French samples, with no observed unsafe private-read release. These are aggregate comparisons across language samples, not within-language estimates; sample composition and recognition conditions are not matched across languages.

narrows the setting to identity-gated private reads, where the relevant speech question is whether the spoken identifier survives transcription exactly enough to authorize disclosure.

D.3 Validator Mechanisms

The ablation and replay results explain where the safety gain comes from. Removing a rule asks how much unsafe release returns when that precondition is no longer enforced. Counterfactual replay starts from unsafe calls and measures which repair stage would have recovered the outcome without executing prematurely.

(a) Validator ablations under perturbation			(b) Counterfactual replay	
Ablation	Safe	Unsafe	Replay statistic	Value
Full VOICEAGENTGUARD	0.822	0.067	Calls already safe	55 / 126
No validator	0.01	0.99	Calls requiring repair	71 / 126
No stability	0.822	0.111	Speech repair gain	0.032
No confirm	0.778	0.111	Entity repair gain	0.063
No auth	0.489	0.400	Tool-argument repair gain	0.063
No handoff	0.644	0.244	Handoff repair gain	0.262

Table A14: Mechanism diagnostics for the guarded validator under perturbation. Panel (a) removes one source of represented evidence at a time; authentication and handoff are the most load-bearing checks, with unsafe release rising to 0.400 without authentication and 0.244 without handoff. Panel (b) replays unsafe calls through counterfactual repair stages; handoff provides the largest single safe-outcome gain because it captures policy cases that should leave automation rather than be completed by another tool call.

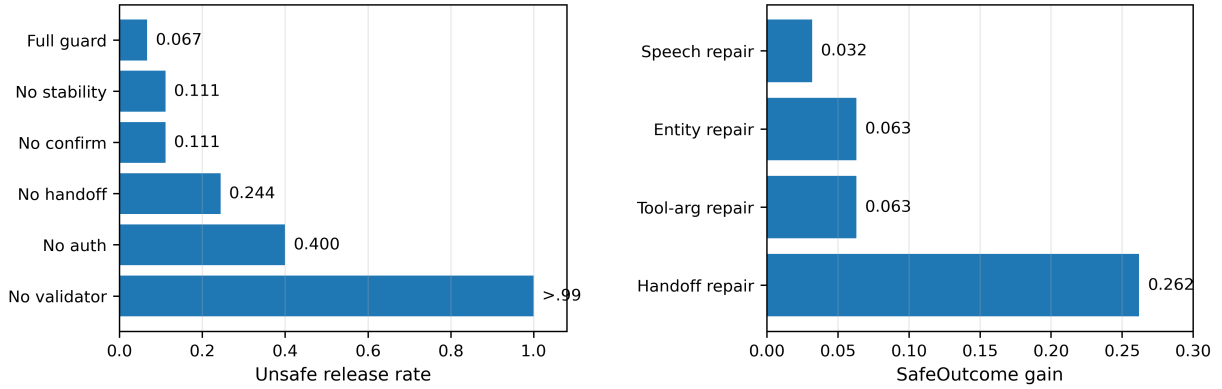


Figure A2: Visual summary of the validator ablation and counterfactual replay diagnostics. The left panel plots unsafe release after removing validator rules; the full guard has unsafe release 0.067, while removing authentication raises unsafe release to 0.400 and removing the validator entirely recovers the permissive baseline. The right panel plots safe-outcome gain from replayed repairs; handoff contributes 0.262 points, larger than entity repair, tool-argument repair, or speech repair in this diagnostic set.

D.4 Planner Fault Injection and Spoken-Identifier Survival

Fault injection tests whether observable planner errors become unsafe releases or non-executing repairs. The identifier-survival analysis then quantifies the speech evidence bottleneck behind the identity-gate replay: exact postcodes are often lost at the turn level, but repeated caller attempts recover some of that evidence.

Err. rate	Base unsafe	Confirm unsafe	Guard unsafe (safe out.)
0.00	0.99	0.742	< 0.01 (0.99)
0.15	0.99	0.742	< 0.01 (0.931)
0.30	0.99	0.742	< 0.01 (0.844)
0.50	0.99	0.742	< 0.01 (0.828)

Table A15: Controlled planner fault injection on the balanced suite ($n=360$ per cell, pooled over three seeds). The injected planner error rate ranges from 0 to 0.50. Base and Confirming continue to release unsupported actions at high rates, while VOICEAGENTGUARD keeps unsafe release below 0.01 by turning represented planner errors into clarification, blocking, or handoff. The decline in guarded safe outcome at higher error rates reflects more repair and non-completion, not unsafe execution.

Recognition/evidence channel	Exact survival	95% CI
ASR 1-best (turn)	0.326	[0.312, 0.340]
ASR n-best (turn)	0.398	[0.383, 0.413]
Any caller attempt (dialogue)	0.579	[0.553, 0.605]

Table A16: Cross-channel exact postcode survival on the English EVI subset ($n=4,152$ turns from 1,384 caller dialogues). Here, channel denotes the evidence path supplied to the identity gate: ASR 1-best, ASR n-best, or any caller attempt accumulated over the dialogue. The results compare aggregate channels rather than outcomes within a channel; they explain the availability gain from retaining additional evidence while the tool remains closed on insufficient identity evidence.

D.5 Illustrative Release Traces

Two traces illustrate the release boundary. In each case, the base agent proposes a plausible tool call without enough represented evidence to authorize it.

ASR correction. A retail caller revises a critical address value after the planner caches it. Base executes the stale argument. VOICEAGENTGUARD marks the entity unstable and requests clarification or confirmation; no mutation occurs until the revision is stable in the ledger.

Authentication failure. A telecom caller requests an account balance after identity verification fails. Base calls `check_balance`; VOICEAGENTGUARD blocks the private read because verified identity is required, making refusal the policy-correct outcome.