The Missing Layer in SOTA AI Controls

A perspective on SOTA AI controls. Anthropic: anthropic.com/engineering/how-we-contain-claude · Google DeepMind: deepmind.google/blog/securing-the-future-of-ai-agents · MACAW: arXiv:2602.10465 · macawsecurity.com/research

Theme	Anthropic identifies	Google DeepMind identifies	MACAW resolution
Detection ceiling	Probabilistic defenses — non-zero miss rate. Model-layer guardrails shape what Claude "tends to do, not what it can do"	Opaque reasoning ceiling — "As models learn to hide their reasoning… simply reading verbalized chain-of-thought will not be enough"	SecCC extracts *effects*, not reasoning. credential_read + data_egress → DENY regardless of model internals or obfuscation. The ceiling disappears when you enforce effects, not behavior.
Speed mismatch	User as injection vector — "when the user is typing the instruction, there's nothing anomalous for a classifier to catch"	Async → sync gap — current system reviews trajectories after the fact; real-time prevention is on the roadmap, not shipped	Every MACAW PEP verification is synchronous — sub-millisecond, inline, before execution. Not a roadmap item. Ships today.
Authorized ≠ safe	Exfiltration through approved domains — destination filter passed an attacker-embedded credential to api.anthropic.com	Insider threat without cryptographic proof — controls are behavioral; a capable insider behaves normally until the moment it acts	Authorization and effect are enforced separately. A valid credential + policy-denied effect = DENY. Every action cryptographically bound to a signed authorization proof.
Alert fatigue at scale	Approval fatigue — users approved ~93% of permission prompts; auto mode catches ~83% of overeager behaviors but ~17% still get through. Any probabilistic defense has a non-zero miss rate	Alert fatigue at scale — most flagged events stem from agent misinterpretation or overeagerness; human review of every trajectory does not scale	Effect-based control is intent-agnostic — whether the agent is misaligned, injected, or overeager, a denied effect combination is blocked automatically. No human review required for the clear-cut cases; policy exceptions are the only escalation path.
Product-scoped containment	Containment is product-specific — claude.ai (server container), Claude Code (local sandbox), Cowork (local VM). A poisoned README loads straight into the model's context despite passing malware checks. No uniform protection across client and server	Internal Google only — roadmap covers GDM internal systems; "model for the wider industry" is aspirational, no external deployment mechanism exists	MACAW protects client and server uniformly — the same cryptographic trust chain secures Claude Code on the developer's laptop and the Databricks MCP server it calls. Signed invocations, policy enforcement, and tamper-evident audit apply at every boundary crossing regardless of where the agent runs.
Multi-agent trust escalation	Multi-agent trust escalation *called out as an open problem — no current solution*	Sandboxing without policy composition — no formal policy algebra governs derived agent actions across multi-step workflows	MAPL composition algebra: four proven theorems — monotonic restriction, transitive denial, bounded derivation, no privilege escalation. A sub-agent cannot escalate above its parent's policy. Not a heuristic.
Persistent memory poisoning	Persistent memory poisoning *called out as an open problem — no current solution*	No instruction provenance — no mechanism to verify lineage through derived prompt chains; behavioral monitoring is bypassable by a capable model	Authenticated prompts carry cryptographic lineage (parent + root + signatures). Context hash-chains make poisoning detectable by construction, not classification. Provenance is structural.