INDEPENDENT RESEARCH · PROJECT SUMMARY

Provable Structural Containment in Multi-Agent AI Systems

Multi-agent AI risk does not live in what any model decides. It lives in the permission structure connecting agents to one another and to privileged actions — and that structure is a graph, checkable before anything is deployed.

Kobe Cargill · Toronto, ON kobesrcargill@gmail.com · (647) 561-6439 · kobecargill.com

THE PROBLEM

In July 2026 Anthropic disclosed that during a security evaluation, one of its models — believing itself sandboxed — published a malicious package to a public registry to win a capture-the-flag task. A security company's scanner installed it automatically, the payload executed, and the model used the resulting credentials to reach that company's infrastructure. **Two organizations with no relationship were connected by a channel neither had drawn**, arising solely from permissions both held on a common public resource. Anthropic's own conclusion was that the failure was closer to a harness and operational problem than a model alignment problem.

The same shape recurs elsewhere: models escaping evaluation sandboxes into production infrastructure by chaining credentials across services, and an agent run against a government ministry with its human-approval step deliberately switched off. In none of these cases was the failure located in an agent's reasoning. It was located in the structure containing it.

THE METHOD

A static, design-time analysis that requires no execution and runs from configuration alone. It (i) reconstructs an agent network's channel graph from **declared permissions**, including the implicit channels created by shared files, memory stores, and tools that appear on no architecture diagram; (ii) **derives** each agent's integrity level from that graph via a taint fixed point rather than requiring hand-assigned trust labels; and (iii) proves whether any untrusted source can reach a privileged action without passing a designated checkpoint.

> When the analysis reports no violation, that holds **exhaustively over every execution** the declared permissions allow. Containment becomes a decidable property of a design rather than a behaviour evaluated after the fact.

PRINCIPAL RESULTS

- **Implicit channels are computable.** With write/read permission matrices `W` and `Rd`, resource-mediated channels are the single product $W \cdot Rd^{\top}$ — surfacing edges no designer drew.
- **A guard is a cut, not a checkpoint.** A sanitizing node removes a source from a protected action if and only if it lies on *every* path from that source — i.e. it is a cut vertex. Guard placement is therefore a graph property, not a matter of policy.
- **Guard counting is computable** (Menger): the number of guards needed equals the number of vertex-disjoint paths, obtained by max-flow — so the cost of governing an architecture is known before it is built.
- **Minimal monitoring placement is tractable.** Because the permission structure is natively bipartite, the minimum sufficient set of monitoring points reduces to bipartite vertex cover and is computable in polynomial time (König).
- **Oversight and integrity share one algorithm.** The condition for an overseer to be able to close a channel is exactly the test for whether that channel is an undesigned backdoor.
- **Integrity degrades monotonically.** Across configuration changes, revoking a permission does not recover the integrity it has already cost — unless the system is *initialisation-robust*, a design property making it certifiable irrespective of its history.

WHY IT MATTERS FOR GOVERNANCE

Most AI governance operates behaviourally and retrospectively: deploy, observe, evaluate. If containment is instead a checkable property of an architecture, a framework can **specify and audit it in advance** — requiring a system to demonstrate coverage of its privileged actions before deployment, and re-demonstrate it on every configuration change. The analysis becomes a gate in the deployment pipeline rather than a periodic audit, which is how safety-critical software has long been certified.

PROGRAMME STATUS

L1	Static Integrity Analysis — base case	COMPLETE
1-R	Reconfiguration: change over time	COMPLETE
1-S	Settling: certifying after a dynamic phase	COMPLETE
L2	Oversight & Control — observability, monitoring placement	COMPLETE
L3	Science of Agent Networks — propagation, centrality, robustness	IN DEV

STATED LIMITS

Sound relative to declared permissions. Channels no declaration captures — covert or out-of-band — are outside any method of this kind.

Certifies placement, not effectiveness. That a guard sits on the path is provable; that it correctly sanitizes is a separate, complementary problem.

Over-approximate by design. False alarms are possible; missed channels are not. This is the correct failure mode for a safety tool.

FOUNDATIONS

Graph theory (reachability, vertex cuts, Menger, König) · order theory (security lattices, Knaster–Tarski and Kleene fixed points) · classical information-flow security (Denning, Biba integrity, non-interference, the confused deputy) · structural control theory (Lin; Liu, Slotine & Barabási).

MATERIALS

Interactive 3D visualisation — the three layers as a multiplex stack.
`interactive-3-d-assets.replit.app`

60-second explainer — animated overview of the problem and method.
`…replit.app/security-explainer-video`

On request: full technical documents for each layer with an interactive walkthrough of the core mechanism; slide deck; and a 29-entry reference glossary linking every theorem to primary sources.

Independent research, unaffiliated with any employer. Positioned against active work in agent information-flow control — Fides (Costa & Köpf et al.), SEAgent, Prompt Flow Integrity — which generally enforce at runtime, on a single planner, or with security labels supplied rather than derived. This work is the static, whole-network, permission-derived analysis. Feedback and collaboration welcome.