Swarm Shepherd: Securing Multi-Agent Ecosystems Against Persistent Latent Compromise

Wuyang Zhang
Department of Computer Science
University of Massachusetts Boston
Boston, MA, USA
doggo@ieee.org

Shichao Pei*
Department of Computer Science
University of Massachusetts Boston
Boston, MA, USA
shichao.pei@umb.edu

Abstract

Agentic ecosystems are emerging as a new platform layer in which agents maintain persistent memory, retrieve from shared knowledge bases, execute through tool interfaces, and coordinate through shared states. These capabilities create a security problem that per-turn defenses miss: compromise can accumulate across sessions, agents, and system layers before visible failure. We define this threat as *Agentic Sleeper*, a system-level adversary that first behaves legitimately, gains routing trust, and later activates across memory, agent logic, and infrastructure. We formulate defense as inference under partial observability and propose **Swarm Shepherd**, a system-level defense framework centered on an inference filter for early warning, paired with a backward trace-back module for entry attribution. Our analysis yields a two-scale deployment rule that balances longer event histories against broader agent coverage. Across LangGraph, CrewAI, and AutoGen, Agentic Sleeper drift begins 2.7–8.9$\times$ faster than natural baselines, propagates across agents on most topologies, and evades per-turn and stateful defenses above 80%. Swarm Shepherd warns at least three sessions before visible degradation (2.6 sessions when failure is scored by the benchmark's own task grader), while trace-back reaches 0.69 top-1 accuracy and 0.72 macro-F1 over 24 variants. The two-scale deployment rule is empirically confirmed, and in live end-to-end runs on LangGraph and AutoGen the posterior-guided defense stack drives cross-agent propagation below the self-sustaining regime, which removing only the visibly compromised agent does not achieve.

1 Introduction

In early 2026, SANDWORM_MODE injected rogue MCP server [1] entries into AI-assistant configurations it found for Claude Code, Claude Desktop, Cursor, VS Code Continue, and Windsurf/Codeium, using trusted tool listings to stage credential exfiltration [2]. A separate LayerX [3] report on Claude Desktop Extensions (DXT) described a CVSS 10/10 zero-click path in which calendar data could be autonomously chained from Google Calendar into a privileged local executor such as Desktop Commander. These incidents reveal an emerging platform layer in which agents persist memory, invoke tools, and update shared state, but enforcement of isolation, per-connector consent, and post-install integrity remains incomplete [3–5].

We call this platform layer the *agentic ecosystem*. Unlike single-turn LLM applications, agentic ecosystems maintain persistent memory, retrieve from shared knowledge bases, execute through tool interfaces such as MCP, and coordinate through shared state. These capabilities change the security problem from detecting isolated unsafe outputs to reasoning about long-horizon risk accumulation: a

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

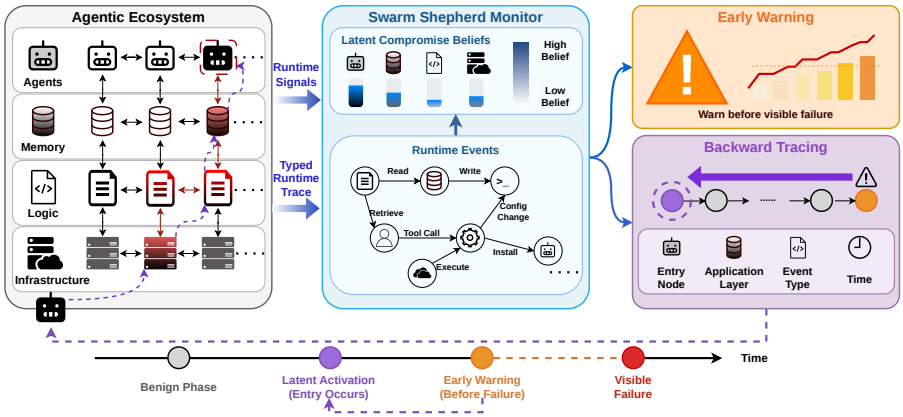

Figure 1: **Swarm Shepherd overview for persistent latent compromise.** Left: an Agentic Sleeper enters as a legitimate agent or tool, then introduces contaminated state that propagates through ordinary memory reads, writes, tool calls, and handoffs. Right: Swarm Shepherd treats typed runtime events as evidence of hidden compromise, maintains cross-layer compromise beliefs, raises early warnings before visible failure, and traces the likely entry node, layer, and time.

benign-looking memory write, retrieval, tool call, or routing decision can persist across sessions, be reused by other agents, and later reshape system behavior.

Existing work primarily frames agentic security as localized attack detection, covering single-interaction prompt-based attacks [6], poisoned memory entries [7], model-level sleepers and backdoors [8, 9], or cross-agent attacks that activate immediately after initial access [10, 11]. This framing leaves open a different threat: compromise that begins from legitimate participation, remains latent over many interactions, and becomes visible only after it has accumulated across sessions, agents, and system layers. We define this missing threat as the *Agentic Sleeper*. An Agentic Sleeper first behaves as a legitimate tool, agent, or handoff target, gradually gaining routing trust through repeated benign interactions. It later converts that position into persistent influence over three ecosystem layers: memory, agent logic, and infrastructure. Memory can store and replay contaminated content, logic can alter routing or tool-selection behavior, and infrastructure can preserve adversarial code or configuration changes across sessions. This distributed attack surface makes compromise only partially visible: a single memory write, retrieval, or tool call may look legitimate, while the attack emerges only when these actions are linked across layers and time.

Defending against Agentic Sleeper requires moving beyond per-action classification. Runtime traces provide observable evidence, including memory writes, retrievals, routing decisions, tool calls, configuration changes, and execution events, but they do not directly identify whether a particular agent, memory store, logic object, or infrastructure component has been compromised. The defender must therefore infer a hidden compromise process from partial observations. We model this problem with two coupled views of the agentic ecosystem: an observable event graph that records typed runtime interactions, and a latent influence graph that tracks how compromise accumulates and propagates across system layers. Together, these views define a factored Partially Observable Markov Decision Process [12–14], where compromise is partially observable through runtime traces and the hidden state is factored across system layers. This formulation turns defense from deciding whether each observed action is malicious into estimating hidden compromise under partial observability.

We instantiate this formulation with **Swarm Shepherd**, a system-level defense framework for *early warning, trace-back, and containment-guided deployment* against Agentic Sleeper compromise. Swarm Shepherd uses the observable event graph as a sensor layer and applies an event-triggered factored Kalman filter to maintain beliefs over hidden compromise across agents, memory, logic, and infrastructure. The filter supports early warning by detecting rising compromise belief before visible failure. When a failure or high-confidence warning occurs, Swarm Shepherd runs a backward trace-back procedure over the same event history to identify the most likely entry node, system layer, and time of compromise. Our main theorem connects this inference problem to finite-population propagation: monitoring error depends on both the depth of event history collected for each agent and the breadth of agent coverage across the ecosystem, yielding a two-scale deployment rule that specifies how much history and coverage are needed for effective containment.

We validate both the threat model and the defense framework across LangGraph, CrewAI, and AutoGen, and make four contributions:

- **Threat model.** The Agentic Sleeper (Section 3.1) produces adversarial drift 2.7 to 8.9× earlier than a 73-interaction natural-drift baseline [15], propagates across agents with $R_0 > 1$[2] on all but the most constrained topologies, and evades the evaluated per-turn guardrails and the stateful DeepContext detector [16] above 80% (Section 5.1).
- **Early warning and trace-back.** Calibrated with split conformal prediction [17], Swarm Shepherd warns 3.2 to 3.9 sessions before failure defined by the Agent Stability Index, and 2.6 sessions (95% CI $[2.1, 3.1]$) before failure scored by the benchmark's own grader; trace-back reaches 0.69 top-1 accuracy and 0.72 macro-F1 over 24 attack variants (Sections 5.2–5.3).
- **Deployment rule.** A two-scale deployment theorem gives a linear boundary between observation depth and agent coverage, and the empirical fit $C_{\text{emp}} = 8.31$ (95% CI $[7.42, 9.18]$) matches the plug-in value 8.3 (Sections 4.5 and 5.4).
- **Measured containment.** In live end-to-end runs, the posterior-guided defense stack holds R_c at 0.81 $[0.64, 0.97]$ on LangGraph star and 0.62 $[0.47, 0.81]$ on AutoGen mesh, whereas removing only the visibly infected agent leaves 2.83 $[2.44, 3.23]$ on LangGraph star (Section 5.5).

2 Related Work

Existing work exposes several ingredients of long-horizon agentic compromise, but studies them largely in isolation. Agent-drift work shows that long-running agents can deviate semantically, coordinatively, or behaviorally from intended objectives [15]; goal-drift and context-drift studies further connect these failures to pattern-matching pressure and stable multi-turn equilibria [18, 19]. These studies establish long-horizon behavioral change, but not an adversary that first builds routing trust and later converts that position into cross-agent compromise. Strategic-corruption and sleeper-agent work provide a second ingredient: online learners can face strategically corrupted feedback under bounded adversarial budgets [20–22], and hidden LLM behaviors can survive safety training or evaluation before activating under delayed or contextual triggers [8, 9]. These lines capture strategic corruption or latent activation, but not persistence across shared memory, logic, and infrastructure. Prompt injection and multi-turn jailbreaks explain local manipulation [6, 23, 24], while memory and cross-agent attacks show that compromise can outlive a single model call through infectious jailbreaks, self-replicating prompts, poisoned experience retrieval, query-only memory injection, and persistent-memory exploitation [10, 11, 25–27]. However, these attacks typically activate immediately or remain concentrated in one layer, such as prompt context, memory retrieval, or self-replicating communication. Together, prior work explains drift, latent activation, local manipulation, and cross-agent spread, but not their composition in a persistent agentic ecosystem where a legitimate-looking agent first builds trust, then activates gradually, persists across sessions, and propagates through memory, logic, and infrastructure while remaining only partially visible through runtime traces.

The defenses closest to our setting fall into stateful monitoring, agentic-security governance, and attribution. Stateful monitors such as DeepContext move beyond per-turn filtering by tracking adversarial intent across a conversation [16], and recent agentic-security surveys emphasize governance controls including input validation, tool-call guards, sandboxing, audit trails, and anomaly detection [28]. These families improve monitoring and governance, but they do not maintain cross-layer beliefs over latent compromise that persists across layers. Attribution-oriented work provides another partial line: WhichAgent attributes task failures to a responsible agent and decisive step in multi-agent execution logs [29], while ProvSEEK and CyberSleuth use provenance or forensic evidence to investigate security incidents after compromise [30, 31]. However, existing defenses do not jointly answer when compromise begins before visible failure, which node and layer introduced it, and how containment should clear persistent residue in an agentic ecosystem. Appendix B.1 gives the positioning against the closest conceptual and theory-facing peers.

[2] R_0 is the standard epidemiological basic reproduction number, adapted here as the expected number of additional agents exposed or compromised by one compromised agent before containment. With a defense in place, the same quantity is the controlled reproduction number R_c.

3 Preliminaries

Definition 1 (Agentic Ecosystem)**.** *An agentic ecosystem is a persistent system of interacting agents, shared memory stores, logic objects, and infrastructure objects, denoted by* $\mathcal{E} = (\mathcal{A}, \mathcal{M}, \mathcal{L}, \mathcal{I})$*. Here,* $\mathcal{A}$ *denotes agents,* $\mathcal{M}$ *denotes memory stores,* $\mathcal{L}$ *denotes logic objects such as system prompts, tool schemas, configurations, and routing policies, and* $\mathcal{I}$ *denotes infrastructure objects such as runtimes, package interfaces, permission boundaries, and service gateways.*

Three-Level Attack Surface The attack surface spans three persistent layers of an agentic ecosystem: memory, logic, and infrastructure (full deployment-interface matrix in Appendix B.14). **(L1) Memory** includes shared knowledge stores that agents read from and write to across sessions with no per-agent access control. Because memory is shared and persistent, L1 compromise can propagate indirectly when other agents reuse the poisoned content. **(L2) Agent Logic** includes agent configurations, tool definitions, system prompts, and source code, stored as plaintext with no integrity verification. L2 compromise is more structural because it changes future decision processes rather than only stored content. **(L3) Infrastructure** includes deployment frameworks, serving stacks, and package-manager interfaces, granting user-level permissions by default. L3 exploits the execution layer that grants agents operational capability. Together, these three layers define the persistent substrate of agentic compromise: memory determines what agents recall, logic determines how agents decide, and infrastructure determines what agents can execute. L1 attacks tend to be retrieval-dependent, while L2 and L3 attacks often produce more deterministic activation through configuration, code, or execution changes.

3.1 Threat Model

Agentic Sleeper. We study a system-level adversary that enters an agentic ecosystem as a legitimate-looking tool-class agent. The adversary first behaves usefully, accumulates routing trust through normal interactions, and later activates to introduce persistent compromise. The threat is therefore not a single unsafe response, but a long-horizon process that changes system state over time.

Attacker goal and knowledge. The attacker seeks persistent ecosystem compromise without triggering existing defenses. Its objectives may include behavioral control, unauthorized tool use, data exfiltration, and configuration manipulation. We assume the attacker has gray-box knowledge of the host OS, agentic deployment patterns, tool interfaces, memory stores, and configuration formats, but does not require access to model weights or internal safety classifiers. After activation, the adversary may influence other agents or ecosystem components. The evaluated Eve implementation follows this threat model by first building routing trust as a legitimate tool-class agent and then activating through L2 configuration writes, L1 memory poisoning, and L3 routing modification. Appendix C gives the phase-by-phase implementation, and Appendix C.8 gives the 24-variant matrix used in evaluation.

Success criteria. An agent is compromised when any of the four triggers fires. **T1 Behavioral drift**: outputs drift from original objectives across sessions without explicit instruction. **T2 Policy violation**: a forbidden operation such as tool install, configuration modification, or out-of-scope exfiltration. **T3 Integrity fault**: a code, configuration, or tool-artifact hash mismatches the trusted baseline. **T4 Task failure**: task-success drops below a task-specific threshold. We measure behavior with the Agent Stability Index (ASI) [15], a composite of behavioral-consistency subscales in $[0, 1]$, and call the first session at which a node's ASI falls below 0.7 its *visible failure*; *warning lead* is the number of sessions from the defender's first warning to this failure. Because ASI subscales also enter the defender's observations (Section 4), we additionally report lead against failure scored by each benchmark's own task grader. At the ecosystem level, an attack succeeds when compromise becomes self-sustaining, measured by an ecosystem reproduction factor $R_0 > 1$, meaning that each compromised agent exposes or compromises more than one additional agent before containment.

4 Swarm Shepherd

Swarm Shepherd formulates agentic-ecosystem defense as inference over observable runtime events and hidden compromise states. It supports early warning, backward trace-back, and ecosystem-level deployment decisions for containment.

4.1 Observable Events and Latent States

Swarm Shepherd represents an agentic ecosystem through two coupled views: an observable event graph that records typed runtime interactions, and a latent state graph that tracks hidden compromise-relevant state on the same nodes. The full logging taxonomy appears in Appendix B.2; the main text keeps the definitions needed for inference, attribution, and deployment.

Definition 2 (Observable Event Graph)**.** *Given an ecosystem $\mathcal{E}$, the observable event graph at time t is the heterogeneous multigraph $\mathcal{G}^{\mathrm{obs}}_{\leq t} = (V^{\mathrm{obs}}, E^{\mathrm{obs}}_{\leq t}, \phi, \psi)$, where $V^{\mathrm{obs}} = \mathcal{A} \cup \mathcal{M} \cup \mathcal{L} \cup \mathcal{I}$, ϕ maps nodes to agent, memory, logic, or infrastructure types, and ψ maps timestamped edges to runtime event types. Memory events include $\{\mathsf{read}, \mathsf{write}, \mathsf{retrieve}\}$, logic events include $\{\mathsf{invoke}, \mathsf{return}, \mathsf{modify}, \mathsf{handoff}\}$, and infrastructure events include $\{\mathsf{grant}, \mathsf{install}, \mathsf{execute}, \mathsf{egress}\}$.*

Definition 3 (Latent State Graph)**.** *The latent state graph is $\mathcal{G}^{\mathrm{lat}}_{\leq t} = (V^{\mathrm{lat}}, E^{\mathrm{lat}}_{\leq t}, Z_t)$, where $V^{\mathrm{lat}} = V^{\mathrm{obs}}$ and $E^{\mathrm{lat}}_{\leq t}$ is induced by the recorded runtime events in $\mathcal{G}^{\mathrm{obs}}_{\leq t}$. Each node v carries a six-dimensional latent compromise state $z_{v,t} = (c_{v,t}, s_{v,t}, a_{v,t}, p_{v,t}, d_{v,t}, x_{v,t}) \in \mathbb{R}^6$, whose coordinates denote contamination, susceptibility, authority, persistence, deviation, and capability exposure. These coordinates define a fixed latent state schema shared across nodes, while $z_{v,t}$ itself is an unobserved random variable. The joint latent state is $Z_t = \{z_{v,t}\}_{v \in V^{\mathrm{lat}}}$. Each typed event $(u, v, r, \tau) \in E^{\mathrm{lat}}_{\leq t}$ is associated with an influence operator $B^{(r,\phi(u),\phi(v))}$, which specifies how event type r transfers compromise-relevant influence from the source node type $\phi(u)$ to the target node type $\phi(v)$. Swarm Shepherd does not assign latent states directly; instead, it maintains posterior beliefs over $z_{v,t}$ from typed events and runtime signals.*

Definition 4 (Factored State-Space Model)**.** *Latent states z evolve as*

$$z_{v,t+1} = A^{\phi(v)} z_{v,t} + \sum_{(u,v,r) \in E^{\mathrm{obs}}_t} B^{(r,\phi(u),\phi(v))} z_{u,t} + C\,\iota_{v,t} + w_{v,t}, \quad w_{v,t} \sim \mathcal{N}(0, Q^{\phi(v)}), \tag{1}$$

and runtime signals are emitted as:

$$y_{v,t} = H z_{v,t} + \eta_{v,t}, \qquad \eta_{v,t} \sim \mathcal{N}(0, \Sigma_{\mathrm{obs}}). \tag{2}$$

Here $A^{\phi(v)}$ captures node-type self-dynamics, $B^{(r,\phi(u),\phi(v))}$ is the typed cross-node influence operator from event (u, v, r, τ), $C\iota_{v,t}$ is an observed defender intervention, and H maps latent state to anchored runtime evidence. The parameters in $\theta = (A^{\phi}, B^{(r,\cdot,\cdot)}, H, Q^{\phi}, \Sigma_{\mathrm{obs}})$ are calibrated before deployment using the pipeline in Appendix B.15; C is set from deterministic intervention semantics.

4.2 Early Warning via Online Belief Inference

Definition 4 specifies how latent compromise evolves, but the defender does not observe Z_t directly. Swarm Shepherd therefore turns typed runtime events and runtime signals into online posterior beliefs over each node state. It observes $\mathcal{G}^{\mathrm{obs}}_{\leq t}$ and runtime signals $y_{\leq t}$, and maintains a factored Gaussian belief $b_{v,t}(z) = \mathcal{N}(z; \mu_{v,t}, \Sigma_{v,t}) \approx p(z_{v,t} \mid \mathcal{G}^{\mathrm{obs}}_{\leq t}, y_{\leq t})$ for each node. Updates occur only on active nodes $\mathcal{A}_t = \{v : E_t(v) \neq \emptyset \text{ or } y_{v,t} \text{ arrives or } \iota_{v,t-1} \neq 0\}$ where $E_t(v) = \{(u, v, r) \in E^{\mathrm{obs}}_t\}$ is the set of incoming events at v. Nodes outside $\mathcal{A}_t$ advance lazily by self-dynamics until they next receive an event or observation. Swarm Shepherd performs an event-triggered Kalman filtering update on these active nodes. The *prediction step* propagates each node's previous belief through its self-dynamics and incoming typed influences, producing a prior belief before observing the current runtime signal. The *correction step* then incorporates any runtime observation to obtain the posterior belief used for warning.

Prediction. For each active node $v \in \mathcal{A}_t$, we aggregate incoming typed events directly into the predicted belief as:

$$\begin{aligned}
\mu^{-}_{v,t} &= A^{\phi(v)} \mu_{v,t-1} + \sum_{(u,v,r) \in E_t(v)} B^{(r,\phi(u),\phi(v))} \mu_{u,t-1} + C\iota_{v,t-1}, \\
\Sigma^{-}_{v,t} &= A^{\phi(v)} \Sigma_{v,t-1} (A^{\phi(v)})^{\top} + \sum_{(u,v,r) \in E_t(v)} B^{(r,\phi(u),\phi(v))} \Sigma_{u,t-1} \big(B^{(r,\phi(u),\phi(v))}\big)^{\top} + Q^{\phi(v)}.
\end{aligned} \tag{3}$$

Correction. When an anchored observation $y_{v,t}$ arrives, the standard Kalman correction applies:

$$K_{v,t} = \Sigma^-_{v,t} H^\top \big(H\Sigma^-_{v,t} H^\top + \Sigma_{\text{obs}}\big)^{-1}, \tag{4}$$
$$\mu_{v,t} = \mu^-_{v,t} + K_{v,t}\big(y_{v,t} - H\mu^-_{v,t}\big), \qquad \Sigma_{v,t} = (I - K_{v,t}H)\Sigma^-_{v,t}.$$

Warning score. The posterior belief $b_{v,t}$ induces a scalar compromise score through $\xi_{v,t} = e^\top z_{v,t}$, where $e \in \mathbb{R}^6$ is a fixed unit aggregation vector over the ordered coordinates (c, s, a, p, d, x). We place its largest weights on contamination c and deviation d because these coordinates mark latent corruption and behavioral drift, while the remaining coordinates supply supporting authority, persistence, and capability-exposure evidence. Rather than declaring a warning from a single observed event, Swarm Shepherd uses the posterior probability that the hidden compromise level has crossed an early-warning threshold:

$$s_{v,t} = \Pr_{z\sim b_{v,t}}\left[e^\top z > \tau_E\right], \tag{5}$$

It raises an early warning for node v at $\hat{T}_v = \min\{t : s_{v,t} \geq \gamma_{\text{warn}}\}$, where γ_{warn} is the calibrated warning-probability threshold. Thus, a warning reflects accumulated evidence from typed events and runtime signals, rather than any individual memory write, retrieval, or tool call. Theorem 1 (Appendix A.3) shows that the posterior error decreases at the $n^{-1/2}$ rate under (A1) linear-Gaussian factored dynamics, (A2) rank-d anchored observability, (A3) bounded cross-node coupling, (A4) a minimum incident-event rate, and (A7) stable self-dynamics; trace-back consistency adds (A5) operator distinguishability, and the population results add exchangeability and bounded threshold density (Appendix B.9). Corollary 2 turns this rate into a k-session warning certificate, which we evaluate at $k = 3$ against a 90% conformal target (Section 5.2).

Cost. With per-node latent dimension $d = 6$, one active-node update costs $O(d^3 + |E_t(v)|d^2)$; the cubic term is in d, not in the number of agents, since the factored belief never forms the dense covariance over all nodes. Per-event cost is therefore near-constant in N under bounded in-degree and a bounded active fraction, and measured scaling to $N = 5{,}000$ confirms this (Appendix E.9).

4.3 Entry Attribution via Backward Trace-Back

After an early warning or visible failure occurs at node v at time t, entry attribution turns this trigger into a backward query: which node introduced the compromise, on which ecosystem layer, and when? Swarm Shepherd searches for an entry triple (u, ℓ, t_0), where u is the candidate entry node, $\ell \in \{\text{L1}, \text{L2}, \text{L3}\}$ is the entry layer, and $t_0 \leq t$ is the candidate entry time. Rather than searching the full ecosystem, it restricts attribution to the upstream neighborhood of the triggered node:

$$\mathcal{E}_v = \{(u, \ell, t_0) : u \in \mathcal{N}_k(v; \mathcal{G}^{\text{obs}}_{\leq t}),\ \ell \in \{\text{L1}, \text{L2}, \text{L3}\},\ t_0 \leq t\}, \tag{6}$$

where $\mathcal{N}_k(v; \mathcal{G}^{\text{obs}}_{\leq t})$ is the k-hop in-neighborhood of v on the observable event graph.

Smoothed beliefs. For each candidate entry (u, ℓ, t_0), Swarm Shepherd computes a smoothed posterior $\tilde{b}_{u,t_0}(z) = p(z_{u,t_0} \mid \omega, y_{\leq t}, \mathcal{G}^{\text{obs}}_{\leq t})$, which estimates the hidden state of candidate node u at the candidate entry time t_0, after conditioning on the later warning or failure trigger $\omega = (v, t)$. Intuitively, this backward pass asks whether the later trigger can be explained by compromise that was already present at u at time t_0. We apply an RTS-style backward smoother [32] only on the k-hop upstream candidate neighborhood; Appendix B.19 gives the local recursion.

Layer-specific patterns. Each layer defines a compromise pattern $\mathcal{C}_\ell$ on the smoothed state and nearby event evidence. For L1 memory-layer entry, $\mathcal{C}_{\text{L1}}$ captures elevated contamination and persistence, typically supported by a `write` to memory followed by later `retrieve` events. For L2 logic-layer entry, $\mathcal{C}_{\text{L2}}$ captures elevated deviation and authority shift, supported by a `modify` event on a logic or configuration object without a matching trusted installation. For L3 infrastructure-layer entry, $\mathcal{C}_{\text{L3}}$ captures elevated capability exposure or persistence, supported by `grant`, `install`, `execute`, or `egress` events, especially when permissions increase.

Entry scoring. Each candidate entry is scored by combining its smoothed posterior evidence with a simple candidate prior:

$$q(u, \ell, t_0) \propto \Pr_{z\sim\tilde{b}_{u,t_0}}[z \in \mathcal{C}_\ell] \cdot p_0(u, \ell, t_0). \tag{7}$$

The first term measures how well the candidate's smoothed latent state matches the layer-specific compromise pattern. The prior $p_0(u, \ell, t_0)$ weakly favors candidates that are more exposed, temporally closer to the trigger, and supported by layer-consistent event evidence; Appendix B.19 gives the exact factorization. Swarm Shepherd outputs the highest-scoring entry triple:

$$(\hat{u}, \hat{\ell}, \hat{t}_0) = \arg \max_{(u,\ell,t_0) \in \mathcal{E}_v} q(u, \ell, t_0). \tag{8}$$

Proposition 3 (Appendix A.3) shows that this estimator is consistent when forward beliefs are accurate and typed operators are distinguishable on anchored coordinates.

Belief-to-action bridge. The trace-back output identifies which persistent substrate should be inspected or contained. L1 attribution triggers memory sanitization and quarantine of retrieved entries over $[\hat{t}_0, t]$. L2 attribution triggers configuration freeze, tool-schema verification, rollback, and hash checks. L3 attribution triggers grant revocation, runtime isolation, egress blocking, and package-state rebuild. The next section aggregates the resulting posterior beliefs into an ecosystem-level propagation model to determine whether containment is sufficient at the population level.

4.4 Ecosystem-Level Propagation

Warning and trace-back operate at node level; deployment also needs a population view. After an Agentic Sleeper enters, compromise can continue to spread through ordinary retrieval, handoff, modification, grant, install, and execution paths, even if the initial entry node is removed. We therefore summarize the posterior beliefs into a coarse propagation view that asks whether compromise is self-sustaining at the ecosystem level. For each node, let $\xi_{v,t} = e^\top z_{v,t}$ be the scalar compromise score used by the early warning. We threshold it into SEIR-like compartments [33]:

$$\begin{aligned} S_t &= \{v : \xi_{v,t} < \tau_E\}, & E_t &= \{v : \tau_E \le \xi_{v,t} < \tau_I\}, \\ I_t &= \{v : \xi_{v,t} \ge \tau_I\}, & R_t &= \{v : \text{contained}\}. \end{aligned} \tag{9}$$

Here S_t contains nodes with no warning-level evidence, E_t contains nodes with early latent compromise, I_t contains nodes whose compromise is treated as operationally harmful, and R_t contains nodes that have been isolated, repaired, or otherwise removed from propagation. The threshold τ_E is the early-warning threshold and τ_I is the infected-compromise threshold.

This compartment view lets us estimate whether compromise will keep spreading. Let β denote the effective propagation rate: how often typed cross-agent contacts move nodes from S_t to E_t. Let γ denote the effective detection-and-containment rate: how quickly warnings and layer-specific interventions move nodes from I_t to R_t. Both rates are estimated from calibration traces. Thus, β summarizes the speed of compromise spread, while γ summarizes the speed of defensive containment. We use the ecosystem reproduction factor

$$R_0 = \beta/\gamma \tag{10}$$

as the population-level containment criterion. Without intervention this is the basic reproduction number R_0; with a defense in place, the same ratio evaluated under the defended rates is the controlled reproduction number R_c, and containment means $R_c < 1$. When the reproduction number exceeds 1, propagation dominates containment: each compromised node is expected to expose or compromise more than one additional node before it is contained. Below 1, containment dominates propagation and the compromise process decays. This gives Swarm Shepherd a clear containment target: use early warnings and trace-back to reduce effective propagation below the self-sustaining regime, rather than only removing the visibly compromised agent. Appendix B.20 gives the full SEIR-style dynamics and turnover-extended reproduction factor; Appendix A.2 gives the finite-population concentration result used in the deployment analysis.

4.5 Deployment: Observation Depth versus Coverage

The final deployment question is whether to collect longer histories per node or monitor more nodes. These two choices reduce different uncertainties: longer histories improve node-level posterior inference, while broader coverage improves ecosystem-level propagation estimation.

Theorem 7 (proof in Appendix A.4) formalizes this tradeoff. With n observed events per monitored node, N monitored nodes, $\hat{I}_N(t)$ the fraction of nodes whose filtered compromise score exceeds τ_I,

and $I(t)$ the population-level SEIR solution, the end-to-end error splits into a filtering term in n and a mean-field term in N, which balance on a linear depth-versus-coverage boundary:

$$\mathbb{E}\big|\hat{I}_N(t) - I(t)\big| \le \underbrace{\frac{\rho_{\max} C_1 L^k}{\nu_{\min}\sqrt{n}}}_{\text{filtering}} + \underbrace{\frac{C_2(\mathsf{H})}{\sqrt{N}}}_{\text{mean-field}}, \qquad n_\circ(N) = C_\circ N. \tag{11}$$

Here L bounds the typed coupling, k is the belief-relevant hop radius, $\nu_{\min}$ is the anchored-observability margin, and $C_2(\mathsf{H})$ depends on the horizon. Below this boundary means $n < n_\circ(N)$: for the current number of monitored nodes, each node has too little event history, so monitoring is inference-limited and collecting longer traces helps most. Above this boundary means $n > n_\circ(N)$: node-level beliefs are already stable enough, so monitoring becomes coverage-limited and adding more monitored nodes helps most.

For our calibrated setting, the plug-in constant is $C_\circ \approx 8.3$. The independent heatmap fit gives $C_{\text{emp}} = 8.31$ (95% CI $[7.42, 9.18]$), which agrees with the plug-in value from Theorem 7 within CI (Section 5.4). Thus, a 20-agent deployment reaches the transition point at roughly 170 observed events per agent. This rule connects inference to containment.

5 Evaluation

We evaluate Swarm Shepherd along the **WHEN / WHICH / HOW** spine: when compromise crosses the warning threshold, which node and layer it entered through, and how it propagates through the ecosystem. For threat-validation and defense-comparison experiments, three frameworks (LangGraph, CrewAI, AutoGen) run four reference topologies (pipeline-3, star-5, mesh-7, Barabási-Albert-20) on TheAgentCompany [34], AgentDojo [35], Agent Security Bench [36], and MultiAgentBench [37]. The scaling experiments in Section 5.4 instead use size-parameterized mesh, star, Barabási-Albert, and hub-and-spoke families to estimate deployment boundaries. The headline victim/attacker portfolio uses four role-bearing LLM backends: Claude Sonnet 4.6, Claude Opus 4.6, Llama-3.3-70B, and StepFun-3.5-Flash. Statistical protocols, model adapters, graph scale, seeds, and compute/cost accounting are in Appendix D.1. We label results by evidence type: *measured* (real framework processes, tools, stores, and config writes in an isolated testbed), *replay* (the frozen detector re-run on recorded traces, no new model calls), and *model-based* (calibrated Monte Carlo). Among main-text results only the containment forecast of Table 27 is model-based.

5.1 Agentic Sleeper Achieves Gradual, Persistent, and System-Wide Compromise

The threat model is validated by comparing the Agentic Sleeper with one-shot injection, multi-turn jailbreaks, immediate memory poisoning, and Agent Smith-style immediate propagation (Table 1). Only the Sleeper jointly reaches all five agents by session 50, survives session restart, and evades Llama Guard 4 at 88.8%; drift onset is 2.7–$8.9\times$ faster than the 73-interaction natural baseline [15]. Per-turn and stateful monitors in their published operating mode remain observation-local: Llama Guard 4, Prompt Guard 2, NeMo Guardrails, Lakera Guard, and DeepContext detect only 8–16% of Sleeper trials. Progent with drift-extended rules reaches 16–22% (10–14% when its rules are blind to the evaluated variants), and even an attacker-known oracle policy caps at 64.8%, because contamination arriving inside permitted tool calls never trips a tool-gating rule (Appendix E.7, which also ports adapted DRIFT and CaMeL). Removing the Sleeper after Exposed agents exist does not halt propagation, since residue keeps seeding retrievals; containment must clear residue and drive R_c below 1 (Section 5.5).

Table 1: Attack-paradigm comparison (measured). Only the Sleeper achieves high stealth, epidemic reach, and persistence together. Warning lead is Swarm Shepherd's lead in sessions before visible failure for each attack paradigm.

Method	Attack succ. (%)	Evasion (%)	Agents compr.	Persist.	Warn lead (sess.)
One-shot injection	22.3 ± 2.6	18.4 ± 2.4	1	N	0.3 ± 0.2
Multi-turn jailbreak	48.7 ± 3.1	34.2 ± 2.9	2	N	1.1 ± 0.3
Memory poison (immediate)	68.2 ± 2.9	58.7 ± 3.0	3	Y	2.0 ± 0.4
Agent Smith (immediate)	73.4 ± 2.7	55.3 ± 3.1	4	Y	2.4 ± 0.4
Agentic Sleeper	$\mathbf{87.6 \pm 2.0}$	$\mathbf{88.9 \pm 1.9}$	**5**	**Y**	$\mathbf{3.9 \pm 0.4}$

Agents compromised out of 5 in the star topology.

Table 2: Early-warning comparison (measured; 1,000 trials, 95% bootstrap CI). Warning lead is the number of sessions between Swarm Shepherd's first warning and visible failure (ASI < 0.7). Conformal coverage at the 0.90 target [17] is 0.908, 0.892, and 0.876 on the three frameworks (Appendix Table 20).

	Warning lead (sessions, ↑)			
Method	**LangGraph star**	**CrewAI pipeline**	**AutoGen mesh**	**AUROC (LG)**
ASI threshold	-0.3 ± 0.4	-0.1 ± 0.3	-0.4 ± 0.3	0.512 ± 0.038
DeepContext	1.4 ± 0.4	1.2 ± 0.4	1.0 ± 0.4	0.718 ± 0.028
Factored Frontier	2.6 ± 0.4	2.4 ± 0.4	2.2 ± 0.4	0.829 ± 0.022
Particle filter (1k)	2.8 ± 0.4	2.5 ± 0.4	2.3 ± 0.4	0.836 ± 0.021
Swarm Shepherd (ours)	$\mathbf{3.9 \pm 0.4}$	$\mathbf{3.5 \pm 0.4}$	$\mathbf{3.2 \pm 0.4}$	$\mathbf{0.918 \pm 0.017}$
Oracle (full state)	5.4 ± 0.4	4.9 ± 0.4	4.5 ± 0.4	0.962 ± 0.012

5.2 Swarm Shepherd Provides Early Warning Before Visible Failure

To test whether Swarm Shepherd gives early warning before visible task failure, we sweep $n \in \{20, 50, 100, 200, 500, 1000, 2000\}$ at $N = 10$ across LangGraph star, CrewAI pipeline, and AutoGen mesh, comparing Swarm Shepherd against a full-joint posterior oracle and seven baselines including output-only ASI threshold, DeepContext, the Factored Frontier approximation [38, 39], and a 1k-particle filter. The oracle has access to the true latent compromise state at every node and time step, removing partial observability entirely; it represents the ceiling on any filter that must infer from runtime traces alone. Swarm Shepherd reaches 3.9 ± 0.4, 3.5 ± 0.4, and 3.2 ± 0.4 warning-lead sessions on LangGraph, CrewAI, and AutoGen respectively. On LangGraph star it closes 42% of the warning-lead gap and 65% of the AUROC gap between the best non-oracle baseline and the oracle. Conformal coverage meets the 0.90 target within interval on LangGraph and CrewAI but not on AutoGen (0.876 ± 0.020), and the posterior error fits the predicted $n^{-1/2}$ rate (Appendix Figure 4). Table 2 compares Swarm Shepherd against representative baselines; full per-framework AUROC, ECE, and coverage breakdowns are in Appendix Table 20.

Robustness (replay). The headline setting reads the contamination anchor from ground-truth injection tags and shares ASI between observations and the failure label; Appendix E removes each advantage in turn (LangGraph star). With deployment provenance metadata replacing the tags, AUROC falls from 0.918 to 0.871 at 8.4% clean FPR, the loss concentrating on memory-layer entries. With ASI and T4 removed and failure scored by the benchmark grader, the lead is 2.6 $[2.1, 3.1]$ sessions. The method also holds under 20% telemetry loss, held-out entry families (at most 0.080 AUROC loss, though attribution loses more), hidden calibration taint, and discrete, switching, and non-Gaussian model variants (Appendix E).

5.3 Trace-Back Identifies the Compromise Entry Point

The trace-back experiment hides the entry triple $(u^\star, \ell^\star, t_0^\star)$ and asks the smoother to recover it over the 24-variant attack matrix (8 per L1/L2/L3) across three frameworks and 30 seeds per variant. Swarm Shepherd reaches 0.69 top-1, 0.87 top-3, and 0.72 macro-F1, with median entry-time error $|\hat{t}_0 - t_0^\star|$ of 3.2 sessions, beating the strongest non-oracle baseline by 0.12 top-1 and closing two-thirds of the gap to a full-latent trace-back oracle (Appendix Table 15). The main confusions in Figure 2 occur where poisoned memory is retrieved by configurable-prompt agents (L1/L2) or where modify edges target orchestrators that can be interpreted as logic or infrastructure (L2/L3), matching the structural invariants tested in Appendix D.25.

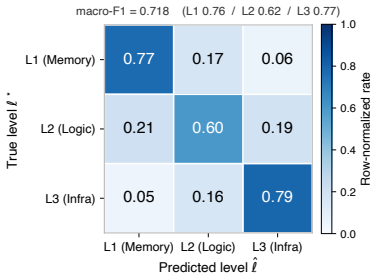

Figure 2: Trace-back entry-layer recovery (measured). Macro-F1 $=$ 0.718 over 24 variants $\times$ 3 frameworks.

5.4 Empirical Scaling Matches the Depth–Coverage Boundary

The mean-field validation verifies the population side of Theorem 4: sweeping $N \in \{7, 20, 50, 100, 500, 1000\}$ across mesh, star, BA, and hub-and-spoke yields $\sqrt{N}$ slopes whose bootstrap CIs cover $-1/2$ except for the expected hub-and-spoke graphon-density penalty (Appendix Figure 5). Composing this population rate with the filter rate over the (n, N) grid across three frameworks and four topologies, the independent heatmap fit $C_{\text{emp}} = 8.31$ $[7.42, 9.18]$ matches the

Table 3: Live end-to-end containment (measured). Controlled reproduction R_c from scorer-verified transmissions with 20 paired Phase-0 snapshots per entry layer; the no-defense rows reproduce Table 31 within CI. Recurrences are counted per run after a full process restart. Full protocol in Appendix E.1.

	LangGraph star (N=5)				AutoGen mesh
Defense arm	R_c **[95% CI]**	**ASR (%)**	**Recurrences**	**Utility loss (%)**	R_c **[95% CI]**
No defense	3.41 [3.06, 3.77]	85.3 ± 3.8	3.2	0	1.34 [1.16, 1.55]
Visible-node removal only	2.83 [2.44, 3.23]	79.1 ± 4.2	2.5	2.1	1.04 [0.84, 1.28]
Warning + quarantine	1.62 [1.32, 1.95]	54.7 ± 4.6	1.0	3.8	—
Full posterior-guided stack	**0.81 [0.64, 0.97]**	**31.2 ± 4.3**	**0.3**	5.6	**0.62 [0.47, 0.81]**

plug-in boundary $n_\circ(N) = 8.3N$ (Appendix Figure 6), putting a 20-agent deployment boundary near 170 events per node.

5.5 Posterior-Guided Defense Contains Spread

The adaptive-stress test applies the Nasr–Carlini [40] adaptive bar to Swarm Shepherd: a white-box adversary can place contamination in $\ker(H)$, dropping T1 detection from the static setting to 0.42 while paying a 34.5% activation-time cost (Appendix Table 16). The key operational lesson is that T1 alone is not enough: null-space attacks still emit modify, install, or task-failure signals, so T2/T3/T4 provide independent evidence on L2/L3 layer compromise.

Containment forecast (model-based). The calibrated forecast fits defense parameters from the preceding measured warning, trace-back, scaling, and adaptive-stress trials, then runs 1,000 Monte Carlo SEIR trials per cell (Appendix B.15). No single defense drives $R_c < 1$ on LangGraph star; even dynamic quarantine leaves $R_c = 1.98$. Fingerprinting plus quarantine nearly contains, and all five paradigms reach $R_c = 0.72$ $[0.64, 0.81]$ while cutting ASR to 28.2% (Table 27).

Measured containment. To test containment without the simulator, every defense arm starts from the same frozen Phase-0 snapshots and faces the live Eve chain; R_c counts only scorer-verified persistent L1/L2/L3 state changes, invisible to the detector, with the estimator behind Table 31 (full protocol in Appendix E.1). The full posterior-guided stack reaches $R_c = 0.81$ $[0.64, 0.97]$ on LangGraph star and 0.62 $[0.47, 0.81]$ on AutoGen mesh, both meeting the pre-declared criterion of an upper 95% bound below 1 (Table 3). Removing only the visibly failed agent does not contain: it leaves 2.5 post-restart recurrences per run against 0.3 for the full stack, because memory, configuration, and package-cache residue survive. The 0.72 forecast is a separate condition and stays labelled a forecast; this run is an independent measurement, not its validation, and the inference stack matters because it conditions both the trigger and the layer-specific containment action on the same posterior belief.

Deployment prescription. LangGraph is most vulnerable (shared state, cycles, checkpoint persistence), CrewAI's Manager is a single point of failure atop shared SQLite3 [41], and AutoGen resists more strongly through higher refusal and sandboxing [42]. The rule is simple: below $n_\circ(N) \approx 8.3N$, lengthen observation windows; above it, broaden coverage. Because trace-back is uncertain, remediation acts on a candidate set: top-3 instead of top-1 cuts recurrence from 0.31 to 0.13 for 1.8 points of utility (Appendix E.8, with the containment playbook and framework hardening).

6 Conclusion

The Agentic Sleeper is a persistent system-level threat in agentic ecosystems, where compromise accumulates across sessions, agents, and system layers before visible failure. We formulated defense as inference under partial observability and introduced Swarm Shepherd, which maintains latent compromise beliefs from typed runtime events and connects early warning, backward trace-back, and ecosystem-level propagation analysis. This shifts defense from isolated action inspection toward long-horizon reasoning over persistent state and cross-layer influence: securing an ecosystem means monitoring not only what an agent does in one interaction, but how ordinary-looking events reshape it over time.

Limitations, impacts, and safeguards. The method rests on a calibrated linear-Gaussian approximation and on benign, stationary calibration telemetry it cannot itself verify, and degrades gracefully but measurably under provenance loss, held-out-family attribution, telemetry loss, and calibration taint (Appendix E); nonlinear filtering, graphon-scale deployment, and the frozen blind red-team test of Appendix E.10 are left open. Dual-use artifacts are sandboxed and gated at release (Appendix B.13).

Acknowledgments and Disclosure of Funding

This work was supported by Coefficient Giving (formerly Open Philanthropy). Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of Coefficient Giving. We also gratefully acknowledge the computing resources and support provided by the National Artificial Intelligence Research Resource (NAIRR) Pilot under Award No. 250100, Modal Labs, Inc, and Google Gemini Academic Program Award.

References

[1] Anthropic. Model context protocol. https://modelcontextprotocol.io/, 2024. Accessed: 2025-01-03.

[2] Socket Threat Research Team. SANDWORM_MODE: Shai-hulud-style npm worm hijacks CI workflows and AI coding assistants. https://socket.dev/blog/sandworm-mode-npm-worm-ai-toolchain-poisoning, February 2026. Accessed: 2026-03-16.

[3] LayerX Security. Claude desktop extensions exposes over 10,000 users to remote code execution vulnerability. https://layerxsecurity.com/blog/claude-desktop-extensions-rce/, February 2026. CVSS 10.0 zero-click RCE via calendar event injection. Accessed: 2026-03-16.

[4] MITRE. CVE-2025-54135: Cursor remote code execution via MCP auto-start (CurXecute). https://nvd.nist.gov/vuln/detail/CVE-2025-54135, 2025. NVD lists NIST CVSS 3.1 9.8 and CNA CVSS 3.1 8.5. Auto-loading MCP entries without user confirmation. Fixed in Cursor 1.3.9. Accessed: 2026-05-04.

[5] MITRE. CVE-2025-54136. https://nvd.nist.gov/vuln/detail/CVE-2025-54136, 2025. Once a user approves a tool configuration, subsequent malicious changes can execute without warning or re-prompt. Fixed in Cursor 1.3. Accessed: 2026-05-04.

[6] Kai Greshake, Sahar Abdelnabi, Shailesh Mishra, Christoph Endres, Thorsten Holz, and Mario Fritz. Not what you've signed up for: Compromising real-world llm-integrated applications with indirect prompt injection. In *Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security*, AISec '23, page 79–90, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400702600. doi: 10.1145/3605764.3623985. URL https://doi.org/10.1145/3605764.3623985.

[7] Balachandra Devarangadi Sunil, Isheeta Sinha, Piyush Maheshwari, Shantanu Todmal, Shreyan Mallik, and Shuchi Mishra. Memory poisoning attack and defense on memory based llm-agents, 2026. URL https://arxiv.org/abs/2601.05504.

[8] Evan Hubinger, Carson Denison, Jesse Mu, Mike Lambert, Meg Tong, Monte MacDiarmid, Tamera Lanham, Daniel M. Ziegler, Tim Maxwell, Newton Cheng, Adam Jermyn, Amanda Askell, Ansh Radhakrishnan, Cem Anil, David Duvenaud, Deep Ganguli, Fazl Barez, Jack Clark, Kamal Ndousse, Kshitij Sachan, Michael Sellitto, Mrinank Sharma, Nova DasSarma, Roger Grosse, Shauna Kravec, Yuntao Bai, Zachary Witten, Marina Favaro, Jan Brauner, Holden Karnofsky, Paul Christiano, Samuel R. Bowman, Logan Graham, Jared Kaplan, Sören Mindermann, Ryan Greenblatt, Buck Shlegeris, Nicholas Schiefer, and Ethan Perez. Sleeper agents: Training deceptive llms that persist through safety training, 2024. URL https://arxiv.org/abs/2401.05566.

[9] Yifei Wang, Dizhan Xue, Shengjie Zhang, and Shengsheng Qian. BadAgent: Inserting and activating backdoor attacks in LLM agents. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9811–9827, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.530. URL https://aclanthology.org/2024.acl-long.530/.

[10] Xiangming Gu, Xiaosen Zheng, Tianyu Pang, Chao Du, Qian Liu, Ye Wang, Jing Jiang, and Min Lin. Agent smith: A single image can jailbreak one million multimodal LLM agents

exponentially fast. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 16647–16672. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/gu24e.html.

[11] Stav Cohen, Ron Bitton, and Ben Nassi. Here comes the ai worm: Preventing the propagation of adversarial self-replicating prompts within genai ecosystems. In *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security*, CCS '25, page 3975–3989, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400715259. doi: 10.1145/3719027.3765196. URL https://doi.org/10.1145/3719027.3765196.

[12] Erik Miehling, Mohammad Rasouli, and Demosthenis Teneketzis. A pomdp approach to the dynamic defense of large-scale cyber networks. *IEEE Transactions on Information Forensics and Security*, 13(10):2490–2505, 2018. doi: 10.1109/TIFS.2018.2819967.

[13] Kim Hammar and Rolf Stadler. Scalable learning of intrusion response through recursive decomposition. In *Decision and Game Theory for Security: 14th International Conference, GameSec 2023, Avignon, France, October 18–20, 2023, Proceedings*, page 172–192, Berlin, Heidelberg, 2023. Springer-Verlag. ISBN 978-3-031-50669-7. doi: 10.1007/978-3-031-50670-3_9. URL https://doi.org/10.1007/978-3-031-50670-3_9.

[14] Maris F.L. Galesloot, Thiago D. Simão, Sebastian Junges, and Nils Jansen. Factored online planning in many-agent pomdps. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'24/IAAI'24/EAAI'24. AAAI Press, 2024. ISBN 978-1-57735-887-9. doi: 10.1609/aaai.v38i16.29689. URL https://doi.org/10.1609/aaai.v38i16.29689.

[15] Abhishek Rath. Agent drift: Quantifying behavioral degradation in multi-agent llm systems over extended interactions, 2026. URL https://arxiv.org/abs/2601.04170.

[16] Justin Albrethsen, Yash Datta, Kunal Kumar, and Sharath Rajasekar. Deepcontext: Stateful real-time detection of multi-turn adversarial intent drift in llms, 2026. URL https://arxiv.org/abs/2602.16935.

[17] Anastasios N. Angelopoulos, Stephen Bates, Adam Fisch, Lihua Lei, and Tal Schuster. Conformal risk control, 2025. URL https://arxiv.org/abs/2208.02814.

[18] Rauno Arike, Elizabeth Donoway, Henning Bartsch, and Marius Hobbhahn. Technical report: Evaluating goal drift in language model agents, 2025. URL https://arxiv.org/abs/2505.02709.

[19] Vardhan Dongre, Ryan A. Rossi, Viet Dac Lai, David Seunghyun Yoon, Dilek Hakkani-Tür, and Trung Bui. Drift no more? context equilibria in multi-turn llm interactions, 2025. URL https://arxiv.org/abs/2510.07777.

[20] Thodoris Lykouris, Vahab Mirrokni, and Renato Paes Leme. Stochastic bandits robust to adversarial corruptions. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, page 114–122, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450355599. doi: 10.1145/3188745.3188918. URL https://doi.org/10.1145/3188745.3188918.

[21] Anupam Gupta, Tomer Koren, and Kunal Talwar. Better algorithms for stochastic bandits with adversarial corruptions. In Alina Beygelzimer and Daniel Hsu, editors, *Proceedings of the Thirty-Second Conference on Learning Theory*, volume 99 of *Proceedings of Machine Learning Research*, pages 1562–1578. PMLR, 25–28 Jun 2019. URL https://proceedings.mlr.press/v99/gupta19a.html.

[22] Tim van Erven, Jack Mayo, Julia Olkhovskaya, and Chen-Yu Wei. An improved algorithm for adversarial linear contextual bandits via reduction. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=MfhFiU28hv.

[23] Mark Russinovich, Ahmed Salem, and Ronen Eldan. Great, now write an article about that: the crescendo multi-turn llm jailbreak attack. In *Proceedings of the 34th USENIX Conference on Security Symposium*, SEC '25, USA, 2025. USENIX Association. ISBN 978-1-939133-52-6.

[24] Andy Zhou and Ron Arel. Tempest: Autonomous multi-turn jailbreaking of large language models with tree search, 2025. URL https://arxiv.org/abs/2503.10619.

[25] Saksham Sahai Srivastava and Haoyu He. Memorygraft: Persistent compromise of llm agents via poisoned experience retrieval, 2025. URL https://arxiv.org/abs/2512.16962.

[26] Shen Dong, Shaochen Xu, Pengfei He, Yige Li, Jiliang Tang, Tianming Liu, Hui Liu, and Zhen Xiang. Memory injection attacks on LLM agents via query-only interaction. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=QINnsnppv8.

[27] Zvika Babo. ZombieAgent: New ChatGPT vulnerabilities let data theft continue (and spread). https://www.radware.com/blog/threat-intelligence/zombieagent/, January 2026. Zero-click indirect prompt injection exploiting ChatGPT memory and connectors for persistent data exfiltration. Accessed: 2026-03-16.

[28] Anshuman Chhabra, Shrestha Datta, Shahriar Kabir Nahin, and Prasant Mohapatra. Agentic ai security: Threats, defenses, evaluation, and open challenges. *IEEE Access*, 14:49455–49482, 2026. ISSN 2169-3536. doi: 10.1109/access.2026.3675554. URL http://dx.doi.org/10.1109/ACCESS.2026.3675554.

[29] Shaokun Zhang, Ming Yin, Jieyu Zhang, Jiale Liu, Zhiguang Han, Jingyang Zhang, Beibin Li, Chi Wang, Huazheng Wang, Yiran Chen, and Qingyun Wu. Which agent causes task failures and when? on automated failure attribution of LLM multi-agent systems. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=GazlTYxZss.

[30] Kunal Mukherjee and Murat Kantarcioglu. Llm-driven provenance forensics for threat investigation and detection, 2025. URL https://arxiv.org/abs/2508.21323.

[31] Stefano Fumero, Kai Huang, Matteo Boffa, Danilo Giordano, Marco Mellia, and Dario Rossi. Cybersleuth: Autonomous blue-team llm agent for web attack forensics, 2026. URL https://arxiv.org/abs/2508.20643.

[32] Herbert E. Rauch, F. Tung, and Charlotte T. Striebel. Maximum likelihood estimates of linear dynamic systems. *AIAA Journal*, 3(8):1445–1450, 1965. doi: 10.2514/3.3166.

[33] P. van den Driessche and James Watmough. Reproduction numbers and sub-threshold endemic equilibria for compartmental models of disease transmission. *Mathematical Biosciences*, 180 (1):29–48, 2002. ISSN 0025-5564. doi: https://doi.org/10.1016/S0025-5564(02)00108-6. URL https://www.sciencedirect.com/science/article/pii/S0025556402001086.

[34] Frank F. Xu, Yufan Song, Boxuan Li, Yuxuan Tang, Kritanjali Jain, Mengxue Bao, Zora Zhiruo Wang, Xuhui Zhou, Zhitong Guo, Murong Cao, Mingyang Yang, Hao Yang Lu, Amaad Martin, Zhe Su, Leander Melroy Maben, Raj Mehta, Wayne Chi, Lawrence Keunho Jang, Yiqing Xie, Shuyan Zhou, and Graham Neubig. Theagentcompany: Benchmarking LLM agents on consequential real world tasks. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2025. URL https://openreview.net/forum?id=LZnKNApvhG.

[35] Edoardo Debenedetti, Jie Zhang, Mislav Balunovic, Luca Beurer-Kellner, Marc Fischer, and Florian Tramèr. Agentdojo: A dynamic environment to evaluate prompt injection attacks and defenses for llm agents. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 82895–82920. Curran Associates, Inc., 2024. doi: 10.52202/079017-2636. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/97091a5177d8dc64b1da8bf3e1f6fb54-Paper-Datasets_and_Benchmarks_Track.pdf.

[36] Hanrong Zhang, Jingyuan Huang, Kai Mei, Yifei Yao, Zhenting Wang, Chenlu Zhan, Hongwei Wang, and Yongfeng Zhang. Agent security bench (ASB): Formalizing and benchmarking attacks and defenses in LLM-based agents. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=V4y0CpX4hK.

[37] Kunlun Zhu, Hongyi Du, Zhaochen Hong, Xiaocheng Yang, Shuyi Guo, Zhe Wang, Zhenhailong Wang, Cheng Qian, Xiangru Tang, Heng Ji, and Jiaxuan You. MultiAgentBench : Evaluating the collaboration and competition of LLM agents. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar, editors, *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8580–8622, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.421. URL https://aclanthology.org/2025.acl-long.421/.

[38] Xavier Boyen and Daphne Koller. Tractable inference for complex stochastic processes. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, UAI'98, page 33–42, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc. ISBN 155860555X.

[39] Kevin Murphy and Yair Weiss. The factored frontier algorithm for approximate inference in dbns. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, UAI'01, page 378–385, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. ISBN 1558608001.

[40] Milad Nasr, Nicholas Carlini, Chawin Sitawarin, Sander V. Schulhoff, Jamie Hayes, Michael Ilie, Juliette Pluto, Shuang Song, Harsh Chaudhari, Ilia Shumailov, Abhradeep Thakurta, Kai Yuanqing Xiao, Andreas Terzis, and Florian Tramèr. The attacker moves second: Stronger adaptive attacks bypass defenses against llm jailbreaks and prompt injections, 2025. URL https://arxiv.org/abs/2510.09023.

[41] Noma Security. Uncrew: High-privilege GitHub token leak in CrewAI (CVSS 9.2). https://noma.security/blog/uncrew-the-risk-behind-a-leaked-internal-github-token-at-crewai/, 2025. CVSS 9.2.

[42] Viet K. Nguyen and Mohammad I. Husain. Penetration testing of agentic ai: A comparative security analysis across models and frameworks, 2025. URL https://arxiv.org/abs/2512.14860.

[43] J.O. Kephart and S.R. White. Directed-graph epidemiological models of computer viruses. In *Proceedings. 1991 IEEE Computer Society Symposium on Research in Security and Privacy*, pages 343–359, 1991. doi: 10.1109/RISP.1991.130801.

[44] Romualdo Pastor-Satorras and Alessandro Vespignani. Epidemic spreading in scale-free networks. *Phys. Rev. Lett.*, 86:3200–3203, Apr 2001. doi: 10.1103/PhysRevLett.86.3200. URL https://link.aps.org/doi/10.1103/PhysRevLett.86.3200.

[45] Cliff Changchun Zou, Weibo Gong, and Don Towsley. Code red worm propagation modeling and analysis. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, CCS '02, page 138–147, New York, NY, USA, 2002. Association for Computing Machinery. ISBN 1581136129. doi: 10.1145/586110.586130. URL https://doi.org/10.1145/586110.586130.

[46] Yen V. Vuong, Maxime Hauray, and Etienne Pardoux. Conditional propagation of chaos in a spatial stochastic epidemic model with common noise, 2021. URL https://arxiv.org/abs/2111.02733.

[47] Alain-Sol Sznitman. Topics in propagation of chaos. In Paul-Louis Hennequin, editor, *Ecole d'Eté de Probabilités de Saint-Flour XIX — 1989*, pages 165–251, Berlin, Heidelberg, 1991. Springer Berlin Heidelberg. ISBN 978-3-540-46319-1.

[48] Thomas G. Kurtz. Solutions of ordinary differential equations as limits of pure jump markov processes. *Journal of Applied Probability*, 7(1):49–58, 1970. doi: 10.2307/3212147.

[49] Nicolas Fournier and Arnaud Guillin. On the rate of convergence in wasserstein distance of the empirical measure, 2013. URL https://arxiv.org/abs/1312.2128.

[50] Anatoli V. Skorokhod. Limit theorems for stochastic processes. *Theory of Probability and Its Applications*, 1:261–290, 1956. URL https://api.semanticscholar.org/CorpusID:122862422.

[51] R.F. Bass. *Diffusions and Elliptic Operators*. Probability and Its Applications. Springer New York, 2006. ISBN 9780387226040. URL https://books.google.com/books?id=tbzgBwAAQBAJ.

[52] Anatoli V. Skorokhod. Limit theorems for stochastic processes. *Theory of Probability and Its Applications*, 1:261–290, 1956. URL https://api.semanticscholar.org/CorpusID:122862422.

[53] Frank Ball, Denis Mollison, and Gianpaolo Scalia-Tomba. Epidemics with two levels of mixing. *The Annals of Applied Probability*, 7(1):46 – 89, 1997. doi: 10.1214/aoap/1034625252. URL https://doi.org/10.1214/aoap/1034625252.

[54] Tom Britton. Stochastic epidemic models: A survey. *Mathematical Biosciences*, 225(1): 24–35, 2010. ISSN 0025-5564. doi: https://doi.org/10.1016/j.mbs.2010.01.006. URL https://www.sciencedirect.com/science/article/pii/S0025556410000143.

[55] Anastasios Tsiamis, Nikolai Matni, and George J. Pappas. Sample complexity of kalman filtering for unknown systems, 2020. URL https://arxiv.org/abs/1912.12309.

[56] Anastasios Tsiamis and George J. Pappas. Finite sample analysis of stochastic system identification. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, page 3648–3654. IEEE Press, 2019. doi: 10.1109/CDC40024.2019.9029499. URL https://doi.org/10.1109/CDC40024.2019.9029499.

[57] Max Simchowitz, Horia Mania, Stephen Tu, Michael I. Jordan, and Benjamin Recht. Learning without mixing: Towards a sharp analysis of linear system identification. In Sébastien Bubeck, Vianney Perchet, and Philippe Rigollet, editors, *Proceedings of the 31st Conference On Learning Theory*, volume 75 of *Proceedings of Machine Learning Research*, pages 439–473. PMLR, 06–09 Jul 2018. URL https://proceedings.mlr.press/v75/simchowitz18a.html.

[58] Andrei N. Borodin and Paavo Salminen. *Handbook of Brownian Motion: Facts and Formulae*. Birkhäuser, Basel, second edition, 2002.

[59] Alexandre B. Tsybakov. *Introduction to Nonparametric Estimation*. Springer Publishing Company, Incorporated, 1st edition, 2008. ISBN 0387790519.

[60] Gal Dalal, Balazs Szorenyi, Gugan Thoppe, and Shie Mannor. Finite sample analyses for td(0) with function approximation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18. AAAI Press, 2018. ISBN 978-1-57735-800-8.

[61] Maxim Kaledin, Eric Moulines, Alexey Naumov, Vladislav Tadic, and Hoi-To Wai. Finite time analysis of linear two-timescale stochastic approximation with markovian noise. In Jacob Abernethy and Shivani Agarwal, editors, *Proceedings of Thirty Third Conference on Learning Theory*, volume 125 of *Proceedings of Machine Learning Research*, pages 2144–2203. PMLR, 09–12 Jul 2020. URL https://proceedings.mlr.press/v125/kaledin20a.html.

[62] Ning Ning and Amin Wu. Bayesian inference for partially observed mckean-vlasov sdes with full distribution dependence, 2026. URL https://arxiv.org/abs/2601.12515.

[63] Sumeet Ramesh Motwani, Mikhail Baranchuk, Martin Strohmeier, Vijay Bolina, Philip H.S. Torr, Lewis Hammond, and Christian Schroeder de Witt. Secret collusion among ai agents: Multi-agent deception via steganography. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 73439–73486. Curran Associates, Inc., 2024. doi: 10.52202/079017-2336. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/861f7dad098aec1c3560fb7add468d41-Paper-Conference.pdf.

[64] Cliff Changchun Zou, Weibo Gong, and Don Towsley. Worm propagation modeling and analysis under dynamic quarantine defense. In *Proceedings of the 2003 ACM Workshop on Rapid Malcode*, WORM '03, page 51–60, New York, NY, USA, 2003. Association for Computing Machinery. ISBN 1581137850. doi: 10.1145/948187.948197. URL https://doi.org/10.1145/948187.948197.

[65] Kai Aizen. AI gateway threat model: 8 attack vectors against LLM gateway infrastructure. https://snailsploit.com/ai-security/ai-gateway-threat-model/, 2026. Accessed: 2026-03-16.

[66] GitHub Advisory Database. CVE-2025-64439: LangGraph Checkpoint affected by RCE in "json" mode of JsonPlusSerializer. https://github.com/advisories/GHSA-wwqv-p2pp-99h5, 2025. Affects langgraph-checkpoint versions before 3.0.0; fixed in 3.0.0. Accessed: 2026-05-04.

[67] LayerX Security. ChatGPT tainted memories: CSRF-based persistent memory injection in ChatGPT Atlas browser. https://layerxsecurity.com/blog/layerx-identifies-vulnerability-in-new-chatgpt-atlas-browser/, October 2025. Persistent memory injection survives across sessions, devices, and tabs. Accessed: 2026-03-16.

[68] OWASP. OWASP top 10 for agentic applications. https://genai.owasp.org/2025/12/09/owasp-top-10-for-agentic-applications-the-benchmark-for-agentic-security-in-the-age-of-autonomous-ai/, December 2025. ASI01: Agent Goal Hijacking is the #1 threat. Accessed: 2026-03-16.

[69] Xianglin Yang, Yufei He, Shuo Ji, Bryan Hooi, and Jin Song Dong. Zombie agents: Persistent control of self-evolving llm agents via self-reinforcing injections, 2026. URL https://arxiv.org/abs/2602.15654.

[70] Wei Zou, Runpeng Geng, Binghui Wang, and Jinyuan Jia. Poisonedrag: knowledge corruption attacks to retrieval-augmented generation of large language models. In *Proceedings of the 34th USENIX Conference on Security Symposium*, SEC '25, USA, 2025. USENIX Association. ISBN 978-1-939133-52-6.

[71] Erfan Entezami and Ali Naseh. Llm misalignment via adversarial rlhf platforms, 2025. URL https://arxiv.org/abs/2503.03039.

[72] Mert Cemri, Melissa Z Pan, Shuyi Yang, Lakshya A Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, Matei Zaharia, Joseph E. Gonzalez, and Ion Stoica. Why do multi-agent LLM systems fail? In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2025. URL https://openreview.net/forum?id=fAjbYBmonr.

[73] GitGuardian. The Nx "s1ngularity" attack: Inside the credential leak that hit 2,349 developers. https://blog.gitguardian.com/the-nx-s1ngularity-attack-inside-the-credential-leak/, August 2025. Compromised npm packages harvested credentials and modified shell configuration files for persistence. Accessed: 2026-03-16.

[74] Truesec. Malicious PyPI package: LiteLLM supply chain compromise. https://www.truesec.com/hub/blog/malicious-pypi-package-litellm-supply-chain-compromise, March 2026. Poisoned PyPI package enabled Kubernetes lateral movement and encrypted credential exfiltration. Accessed: 2026-03-16.

[75] Ajeet Singh Raina. MCP security issues threatening AI infrastructure. https://www.docker.com/blog/mcp-security-issues-threatening-ai-infrastructure/, July 2025. 43% of MCP servers found to have command injection flaws. Accessed: 2026-03-16.

[76] Yechao Zhang, Shiqian Zhao, Jie Zhang, Gelei Deng, Jiawen Zhang, Xiaogeng Liu, Chaowei Xiao, and Tianwei Zhang. Mind your HEARTBEAT! Claw background execution inherently enables silent memory pollution, 2026. URL https://arxiv.org/abs/2603.23064. Zero-click memory pollution via heartbeat background execution achieves 75.6% cross-session behavioral influence. Accessed: 2026-04-01.

[77] MITRE. CVE-2026-25253: OpenClaw one-click remote code execution via WebSocket authentication token exfiltration. https://nvd.nist.gov/vuln/detail/CVE-2026-25253, 2026. CVSS 8.8. Affects OpenClaw versions before 2026.1.29. Accessed: 2026-03-16.

[78] Tianneng Shi, Jingxuan He, Zhun Wang, Linyu Wu, Hongwei Li, Wenbo Guo, and Dawn Song. Progent: Programmable privilege control for llm agents. *CoRR*, abs/2504.11703, April 2025. URL https://doi.org/10.48550/arXiv.2504.11703.

[79] Emanuele Rossi, Ben Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael Bronstein. Temporal graph networks for deep learning on dynamic graphs. In *ICML 2020 Workshop on Graph Representation Learning*, 2020.

[80] Hao Li, Xiaogeng Liu, CHIU Hung Chun, Dianqi Li, Ning Zhang, and Chaowei Xiao. DRIFT: Dynamic rule-based defense with injection isolation for securing LLM agents. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=oY1Xnt83oJ.

[81] Edoardo Debenedetti, Ilia Shumailov, Tianqi Fan, Jamie Hayes, Nicholas Carlini, Daniel Fabian, Christoph Kern, Chongyang Shi, Andreas Terzis, and Florian Tramèr. Defeating prompt injections by design, 2025. URL https://arxiv.org/abs/2503.18813.

Appendices

A Theoretical Proofs and Derivations **21**

Theorem Statements . . . 21

Proof of Theorem 4 (Latent Aggregate Mean-Field Rate) . . . 22

Proof of Theorem 1 (Swarm Shepherd Belief Consistency) . . . 25

Proof of Theorem 7 (Two-Scale Deployment Regime) . . . 26

Proof of Theorem 27 (Optimal Activation Time) . . . 27

SEIR Derivations and Stability Analysis . . . 29

Amplification Results . . . 29

Proposition: Calibration Consistency (Statement and Proof) . . . 30

B Formal Model Details **31**

Main-Text Maps and Positioning . . . 31

Observable Edge Types . . . 31

Shared Latent Coordinates . . . 32

Type-Specific Latent Dimensions . . . 33

Factored DBN and POMDP Formalization . . . 33

Decision Problems: Full Formalizations . . . 34

Identifiability via Anchored Emissions . . . 34

Conformal Risk Control Protocol . . . 35

Assumptions-to-Results Map . . . 35

Operational Interpretation . . . 36

Defender Paradigms: Detailed Action Rules . . . 36

Extended Limitations . . . 37

Disclosure and Release Safeguards . . . 37

Problem-Setting Support Tables . . . 37

Calibration Pipeline and Cold Start . . . 37

Deployment Recipe: Full Integration Detail . . . 39

A5 Hardening Cells . . . 39

Swarm Shepherd Algorithm (Full Pseudocode) . . . 39

Trace-Back Smoother and Candidate Prior . . . 40

SEIR ODE Form, Reproduction Number, and Optimal Activation Time . . . 40

C Attack Lifecycle Detail **41**

Phase 0: Trust-Building in Detail . . . 41

Phases 1 through 4: Detailed Walkthrough . . . 42

Phase 5: Epidemic Propagation in Detail . . . 43

Five Drift Primitives: Formal Definition and Detailed Descriptions . . . 43

Permission Model Analysis . . . 44

Discovery Phase Configuration Paths . . . 44
Deployment Interface Attack Surface Details . . . 45
Attack-Variant Matrix and Root-Cause Signatures . . . 45

D Extended Evaluation Results **46**

Full Setup and Protocol . . . 46
Main-Text Evaluation Evidence . . . 47
Graph Scale and Memory Footprint . . . 47
Swarm Shepherd Ablation Results . . . 48
Progent Drift-Adapted Evaluation . . . 48
Belief-State Intervention (stretch) . . . 50
Per-Model Drift Onset (Extended) . . . 50
Self-Reinforcing Loops . . . 50
SEIR Convergence and Threshold Validation . . . 50
Defense Ablation . . . 51
Adaptive Adversary . . . 51
Infrastructure Feasibility . . . 52
End-to-End Attack Chain . . . 52
Cross-Framework Propagation . . . 52
Per-Variant Trace-Back Results (A4 detail) . . . 53
Multi-Trigger Sensitivity (B6) . . . 53
Warm-up Sweep (B4) . . . 53
Cross-Model Transferability (B5) . . . 54
Adaptive Adversary Detail (A5) . . . 54
Adaptive Attacks against the Five Defense Paradigms (A6 Detail) . . . 55
Per-Primitive Drift Onset . . . 56
Propagation Dynamics by Framework and Topology . . . 57
Benign-Operation Cost and Declaration-Threshold Sweep . . . 57
A3 Mean-Field Rate: Extended Detail . . . 57
A4b Invariant Ablations . . . 58
Framework Vulnerability Matrix . . . 58

E Robustness, Containment, and Fair-Baseline Studies **58**

Measured Live Containment . . . 58
Provenance Quality and Telemetry Degradation (replay) . . . 59
Hidden Calibration Contamination . . . 60
Held-Out Entry-Family Transfer (replay) . . . 60
Model and Emission Sensitivity . . . 61
Parameter Provenance . . . 61
Fairer Baseline Protocols . . . 62
Posterior-Set Remediation and Framework Hardening . . . 62
Measured Scaling . . . 63

Blind Red-Team Protocol 63

A Theoretical Proofs and Derivations

This appendix states and proves the formal guarantees used by Section 4. The three main results (Swarm Shepherd belief consistency, trace-back consistency, and the two-scale deployment theorem) share the same factored state-space model of Definition 4 and the same identifiability backbone $\nu_{\min}$ from Lemma 25.

A.1 Theorem Statements

Theorem 1 (Swarm Shepherd Belief Consistency)**.** *Under (A1) linear-Gaussian factored DBN, (A2) rank-d anchored observability with $\nu_{\min}$, (A3) bounded coupling $\|B^{(r,\cdot,\cdot)}\|_{\mathrm{op}} \le L$, (A4) minimum incident-event rate $\lambda_{\min}$, and stable self-dynamics from (A7) (Appendix B.9), let $b_{v,n}$ be Swarm Shepherd's belief after n events and $b^\star_{v,n}$ the oracle posterior. For $\epsilon, \delta \in (0,1)$,*

$$\Pr\big[\|b_{v,n} - b^\star_{v,n}\|_{\mathrm{TV}} \le \epsilon\big] \ge 1-\delta \quad \textit{whenever} \quad n \ge n_0 = O\!\left(\frac{d^2 L^{2k} \log(d/\delta)}{\epsilon^2 \lambda_{\min}^2 \nu_{\min}^2}\right), \tag{12}$$

where k is the hop radius of v's belief-relevant neighborhood. Proof in Appendix A.3.

Corollary 2 (Time-to-Warning Lead)**.** *Let $\hat{T}_v = \min\{t : b_{v,t}(\xi_{v,t} > \tau_E) > 1/2\}$ be Swarm Shepherd's warning time, $T^\star_{E,v} = \inf\{t : \xi_{v,t} \ge \tau_E\}$ the true early-threshold crossing, and $\tau^\star_v = \inf\{t : \xi_{v,t} \ge \tau_I\}$ the true Infected crossing. Let $\Delta_\tau = \tau_I - \tau_E > 0$. Suppose the TV error event in Theorem 1 localizes the warning time with slack η_ϵ, meaning $\hat{T}_v \le T^\star_{E,v} + \eta_\epsilon$ on that event. Under the (A7) warning-band bounds $|a_t| \le \bar{a}$ and $\sigma_t^2 \le \bar{\sigma}^2$, for any $k \ge 1$ with $\Delta_\tau > \bar{a}(k+\eta_\epsilon)$,*

$$\Pr[\hat{T}_v + k \le \tau^\star_v] \ge 1 - \delta - \min\left\{1,\, 2\exp\left(-\frac{(\Delta_\tau - \bar{a}(k+\eta_\epsilon))^2}{2\bar{\sigma}^2(k+\eta_\epsilon)}\right)\right\}. \tag{13}$$

Thus a k-session warning certificate requires a calibrated threshold margin that dominates both short-horizon drift and diffusion. Proof in Appendix A.3.

Proposition 3 (Trace-Back Consistency)**.** *Under the assumptions of Theorem 1 plus (A5) pairwise operator distinguishability $\min_{r \ne r'} \|B^{(r,\cdot,\cdot)} - B^{(r',\cdot,\cdot)}\|_{\mathrm{op}} \ge \delta_B$ on the anchored subspace,*

$$\Pr[\hat{\ell} = \ell^\star] \ge 1 - \delta - \exp(-c\delta_B^2 n / \nu_{\min}^2), \tag{14}$$

for universal $c > 0$, whenever (12) *is met. The same rate (restricted to the k-hop in-neighborhood of the failure node) governs $\hat{u} = u^\star$. Proof in Appendix A.3.*

Theorem 4 (Latent Aggregate Mean-Field Rate)**.** *Under agent exchangeability, Poisson-driven typed events with absorbing R, (A2.5) Wasserstein-Lipschitz aggregated drift, and stable self-dynamics from (A7), with $\bar{\xi}_N(t) = N^{-1}\sum_v \xi_{v,t}$ and $\xi^\star(t)$ the McKean–Vlasov limit of* (18)*,*

$$\mathbb{E}\big[W_1(\bar{\xi}_N(\cdot), \xi^\star(\cdot))\big] \le \frac{C(\mathsf{H})}{\sqrt{N}}, \tag{15}$$

where $C(\mathsf{H})$ depends only on the horizon, coupling bound L, and kernel Lipschitz constants. Proof in Appendix A.2.

Corollary 5 (SEIR Rate Specialization)**.** *Under bounded density at τ_E, τ_I and the (A7) warning-band variation condition, the thresholded fractions (S, E, I, R) converge to the classical SEIR ODE, with β the spectral radius of $B^{(\mathsf{retrieve},\mathcal{M},\mathcal{A})} + B^{(\mathsf{handoff},\mathcal{A},\mathcal{A})}$ times the typed-event rate, σ^{-1} the first-passage time $\tau_E \to \tau_I$, and γ the b_t-driven detection hazard. This recovers prior malware-propagation models [43–45] as a corollary. Proof in Appendix A.2.*

Remark 6 (Adversary-controlled incubation)**.** *Unlike biological SEIR where σ is pathogen-fixed, σ^{-1} here is a first-passage time set by the attacker's per-contact influence magnitude. Higher aggression gives faster but detectable incubation; distributed drift is slower and stealthier, breaking defenses that rely on fixed incubation.*

Theorem 7 (Two-Scale Deployment Regime)**.** *Under the assumptions of Theorems 1, 4 plus bounded density $\rho_{\max}$ at τ_I, let $\hat{I}_N(t) = N^{-1}\sum_v \mathbb{1}[\hat{\xi}_{v,t} \ge \tau_I]$ and $I(t)$ the SEIR ODE solution. Then*

$$\mathbb{E}\Big|\hat{I}_N(t) - I(t)\Big| \le \underbrace{\rho_{\max} \cdot \frac{C_1 L^k}{\nu_{\min}\sqrt{n}}}_{E_{\mathrm{filter}}(n)} + \underbrace{\frac{C_2(\mathsf{H})}{\sqrt{N}}}_{E_{\mathrm{MF}}(N)}, \tag{16}$$

with $C_1 \le c_1 d(1-\rho(A))^{-1}$, $\rho(A) < 1$ the dominant self-dynamics eigenvalue from (A7); $C_2(\mathsf{H})$ tracked in Appendix A.4, polynomial $c_2(1-L)^{-1}\mathsf{H}^{1/2}$ under (A3) with $L < 1$, worst-case $c_2 e^{L\mathsf{H}}\mathsf{H}^{1/2}$. The two terms equalise at

$$n_\circ(N) = \frac{\rho_{\max}^2 C_1^2 L^{2k}}{\nu_{\min}^2 C_2(\mathsf{H})^2} \cdot N. \tag{17}$$

Residual error is inference-limited *below the boundary (grow n) and* population-limited *above it (grow N). Proof in Appendix A.4.*

A.2 Proof of Theorem 4 (Latent Aggregate Mean-Field Rate)

We prove Theorem 4 on the latent scalar aggregate $\bar{\xi}_N$, then deduce Corollary 5 (the SEIR specialization) by composing with the threshold map. The proof adapts the quantitative coupling template of [46] to our Poisson-driven typed-event construction.

Setup. Let $e \in \mathbb{R}^6$ be the aggregation vector (dominant weight on contamination c and deviation d) and define $\xi_{v,t} = e^\top z_{v,t}$. Projecting the transition of Definition 4 onto e yields the per-agent scalar process

$$d\xi_{v,t} \;=\; \Big[-\kappa_A\, \xi_{v,t} + \sum_r \lambda_{B,r} \sum_{u\,:\,(u,v,r)\in E_t^{\mathrm{obs}}} \xi_{u,t} \Big] dt + \Sigma(\xi_{v,t})\, dW_{v,t}, \tag{18}$$

with $\kappa_A = 1 - e^\top A^{\phi(v)} e/\|e\|^2$ and $\lambda_{B,r} = e^\top B^{(r,\cdot,\cdot)} e/\|e\|^2$. Typed events are driven by independent Poisson processes of rates $\lambda_r(\bar{\mu}_N(t))$ that depend only on the empirical measure $\bar{\mu}_N(t) = N^{-1}\sum_v \delta_{\xi_{v,t}}$.

Standing assumptions for Theorem 4. In addition to the exchangeability and Poisson-driven construction of Definition 4, we adopt (A2.5). The aggregated drift is Lipschitz in the Wasserstein-1 metric, $\|\mathrm{drift}(\mu_1) - \mathrm{drift}(\mu_2)\|_\infty \le L_{\mathrm{drift}}\, W_1(\mu_1,\mu_2)$, with $L_{\mathrm{drift}} = \|\lambda_B\|_\infty \cdot \sum_r \|B^{(r,\cdot,\cdot)}\|_{\mathrm{op}}$ finite by assumption (A3) of Theorem 1. The initial latent distribution has bounded second moment, which propagates to $\sup_{t\in[0,\mathsf{H}]} \mathbb{E}[\xi_{v,t}^2] \le K$ by spectral stability of the self-dynamics (the spectral radius of $A^{\phi(v)}$ is bounded by (A7)).

The proof factors through three internal lemmas that formalize coupling, empirical-measure concentration, and path-space aggregation. Each lemma's conditions are explicitly verified from the standing assumptions.

Lemma 8 (Coupled-SDE Grönwall bound)**.** *Under (A2.5) and the Skorohod coupling of the proof below, the per-agent deviation $\Delta_v(t) = \xi_{v,t} - \tilde{\xi}_{v,t}$ on $[0,T]$ satisfies $\mathbb{E}|\Delta_v(T)| \le e^{(L_{\mathrm{drift}}+\kappa_A)T} \cdot \int_0^T \mathbb{E}[W_1(\bar{\mu}_N(s),\mu^\star(s))]\, ds$.*

Proof. Apply Itô's formula to $|\Delta_v(t)|$ on the coupled SDE. Drift contributions are bounded by $L_{\mathrm{drift}}\, W_1(\bar{\mu}_N(s),\mu^\star(s))$ through (A2.5), and bounded-variation contributions from the identically-thinned Poisson processes vanish in expectation. The integrated differential inequality is $d\,\mathbb{E}|\Delta_v(t)| \le L_{\mathrm{drift}}\, \mathbb{E}[W_1(\bar{\mu}_N(t),\mu^\star(t))]\, dt + \kappa_A\, \mathbb{E}|\Delta_v(t)|\, dt$. Grönwall's inequality on $[0,T]$ yields the claim. □

Lemma 9 (Empirical-measure concentration on the horizon)**.** *Under exchangeability of $(\tilde{\xi}_{v,s})_v$ from the McKean–Vlasov construction and the inherited moment bound $\sup_{s\in[0,\mathsf{H}]} \mathbb{E}[\xi_{\star,s}^2] \le K$, the empirical measure of the i.i.d. copies satisfies $\mathbb{E}[W_1(\bar{\mu}_N^{\mathrm{iid}}(s),\mu^\star(s))] \le C'/\sqrt{N}$ uniformly in $s \in [0,\mathsf{H}]$, and the Skorohod coupling transfers this rate to the true empirical measure $\bar{\mu}_N$ at cost $\mathbb{E}|\Delta_v|$.*

Proof. Propagation of chaos [47, Theorem I.1] and Kurtz's limit [48, Theorem 2.1] give weak convergence of the i.i.d. empirical measure to $\mu^\star(s)$. Under the verified moment bound (inherited from bounded initial second moment and spectral stability of $A^{\phi(v)}$), Fournier and Guillin [49, Theorem 1] provide the W_1 rate $C'/\sqrt{N}$. The coupling of Step 1 below transfers this rate to $\bar{\mu}_N$ by the triangle inequality. □

Lemma 10 (Path-space aggregation)**.** *For the empirical mean $\bar{\xi}_N(t) = N^{-1}\sum_v \xi_{v,t}$, the Kantorovich–Rubinstein duality gives $W_1(\bar{\xi}_N(\cdot), \xi^\star(\cdot)) \le \mathbb{E}|\Delta_v|$, where Δ_v is the per-agent coupling error of Lemma 8 and $\xi^\star$ is the mean-field solution.*

Proof. By Kantorovich duality, $W_1(\bar{\xi}_N, \xi^\star) = \sup_{f\in \mathrm{Lip}_1} |\mathbb{E}f(\bar{\xi}_N) - \mathbb{E}f(\xi^\star)|$. The empirical mean is $1/N$-Lipschitz in the per-agent coordinates, and the McKean–Vlasov copies have $\mathbb{E}\tilde{\xi}_{v,s} = \xi^\star(s)$, so the duality bound contracts to $\mathbb{E}|\Delta_v|$ along any Lipschitz test function. □

Proof of Theorem 4. Step 1 (coupling to i.i.d. copies). Let $(\tilde{\xi}_{v,t})_{v=1}^N$ be i.i.d. copies governed by the McKean–Vlasov limit

$$d\tilde{\xi}_{v,t} = \Big[-\kappa_A\, \tilde{\xi}_{v,t} + \mathrm{drift}^\star(t)\Big]\, dt + \Sigma(\tilde{\xi}_{v,t})\, d\tilde{W}_{v,t},$$

where $\mathrm{drift}^\star(t)$ uses the population-level drift under $\xi^\star$. By Skorohod embedding [50], construct a coupling $(\xi, \tilde{\xi})$ on a common probability space $(\Omega, \mathcal{F}, \mathbb{P})$ driven by the same Brownian motions $(W_{v,t})_v$ and the same thinned Poisson processes $(P_{v,r,t})_{v,r}$. The two processes differ only in the drift evaluation (empirical $\bar{\mu}_N$ versus mean-field $\mu^\star$), and $\bar{\mu}_N(t)$ is $\mathcal{F}_t$-measurable by construction.

Step 2 (coupling-error bound). Lemma 8 applies under (A2.5). Its conclusion, $\mathbb{E}|\Delta_v(T)| \le e^{(L_{\mathrm{drift}}+\kappa_A)T}\int_0^T \mathbb{E}[W_1(\bar{\mu}_N(s), \mu^\star(s))]\, ds$, bounds per-agent deviation by an integrated empirical-measure error.

Step 3 (empirical-measure rate). Lemma 9 applies under exchangeability and the moment bound verified in the setup. Its conclusion, $\mathbb{E}[W_1(\bar{\mu}_N^{\mathrm{iid}}(s), \mu^\star(s))] \le C'/\sqrt{N}$ uniformly on $[0, \mathsf{H}]$, is transferred to $\bar{\mu}_N$ through Step 1's coupling.

Step 4 (path-space assembly). Lemma 10 applies. Chaining Lemmas 8, 9, and 10 gives

$$\mathbb{E}[W_1(\bar{\xi}_N(\cdot), \xi^\star(\cdot))] \;\le\; \frac{C(\mathsf{H})}{\sqrt{N}}, \qquad C(\mathsf{H}) = C' \cdot \mathsf{H} \cdot e^{(L_{\mathrm{drift}}+\kappa_A)\mathsf{H}},$$

with H the time horizon. The exponential dependence on H is the coupling cost of (A2.5) over long horizons and restricts the operational regime as noted in Section 6. □

Lemma 11 (Boundary upcrossing hazard under uniform ellipticity)**.** *Under (A1) linear-Gaussian dynamics, (A3) bounded cross-level coupling, and (A7) bounded warning-band variation, the scalar process $\xi_{v,t}$ of* (18) *is uniformly elliptic and admits a unique, strictly-positive, $C^{1,2}$ transition density on $\mathbb{R}\times(0, \mathsf{H}]$ (Bass [51, Theorem 2.2.1]). For any threshold τ whose marginal density is bounded by $\rho_{\max}$ on a neighborhood $(\tau-\delta, \tau)$, the boundary is transversal (bounded-variation drift plus non-degenerate diffusion), and Nagasawa–Wentzell [52, Proposition I.3.17] gives the upcrossing intensity $\lambda_\tau(t) = \rho(\tau^-; \xi_{\cdot,t}) \cdot [\text{drift toward } \tau]_+$, where $\rho(\tau^-;\cdot)$ is the marginal density immediately below τ.*

Proof. Bass [51, Theorem 2.2.1] gives existence and uniqueness of the $C^{1,2}$ transition density; strict positivity follows from non-degeneracy of the diffusion. Transversality of the threshold is a consequence of the drift being of bounded variation and the diffusion being strictly positive on a neighborhood of τ. Under these conditions, the Nagasawa–Wentzell intensity formula [52, Proposition I.3.17] applies. □

Proof of Corollary 5 (SEIR specialization). The proof identifies the three SEIR rates β, σ, γ as explicit functionals of the scalar process (18), then composes with Theorem 4 on the threshold-map functionals.

Claim 1 (transmission rate β). Apply Lemma 11 at τ_E under the bounded-density assumption of Theorem 7. Substituting the typed-event mean rate c and the aggregated drift coefficients $\lambda_{B,\mathsf{retrieve}}, \lambda_{B,\mathsf{handoff}}$ of (18) into the upcrossing intensity yields $\beta = \rho\,(\lambda_{B,\mathsf{retrieve}} + \lambda_{B,\mathsf{handoff}}) \cdot c_{\max}$.

Claim 2 (incubation rate σ). Define $\sigma^{-1} = \mathbb{E}[\tau_{\mathrm{FP}}(\tau_E \to \tau_I)]$ with τ_{FP} the first-passage time of $\xi_{v,t}$ under (18) from level τ_E to τ_I. Finiteness is a precondition for the reduction, and we verify it here. By (A7), the spectral radius of $A^{\phi(v)}$ is bounded by a constant below 1, so $\kappa_A > 0$ in (18) and the drift is dissipative outside a compact set. Applying the optional stopping theorem to the martingale

derived from Itô's formula on $\xi_{v,t}^2$ gives $\mathbb{E}[\tau_{\mathrm{FP}}] < \infty$, so sojourn in the Exposed compartment is integrable. Kurtz's reduction [48, Theorem 2.1] then applies under its three hypotheses, each verified here: (i) exchangeability by Definition 4, (ii) bounded drift and jump rates by (A2.5), (A3), and (A7), (iii) integrable sojourn time just shown. The reduction gives the deterministic ODE (27) at this σ, independent of the shape of the first-passage distribution. The exponential-tail approximation used by classical SEIR is a separate assumption (small drift-to-variance ratio at the τ_E-boundary) that we neither make nor invoke.

Claim 3 (detection rate γ). Let $T_{\mathrm{det}} = \inf\{t : b_{v,t}(\xi > \tau_I) > 1/2\}$ be the declaration stopping time. Let $(\mathcal{F}_t^+)$ denote the right-continuous completion of the anchored-observation filtration. Every stopping time on $(\mathcal{F}_t^+)$ admits a compensator by Doob–Meyer. We need the stronger statement that the compensator is absolutely continuous with respect to Lebesgue time, so that a hazard rate exists. T_{det} is totally inaccessible on $(\mathcal{F}_t^+)$ because it is driven by the continuous Swarm Shepherd posterior and not by predictable events in the anchored-event filtration. Jacod and Shiryaev [52, Chapter I, Theorem 3.17] therefore give a unique predictable compensator with a density $\gamma(t)$ with respect to Lebesgue time. This γ is the hazard rate $\gamma(t) = \lim_{\Delta t \to 0} \Delta t^{-1} \Pr[T_{\mathrm{det}} \in (t, t+\Delta t] \mid \mathcal{F}_t^+]$ as a functional of the Swarm Shepherd posterior trajectory.

Composition. For each compartment indicator $\mathbb{1}[\xi_{v,t} \in \{S, E, I, R\}]$, Lemma 18 (bounded-density continuity modulus) bounds the classification error at any threshold $\tau \in \{\tau_E, \tau_I\}$ by $\rho_{\max} W_1(\xi_{v,t}, \bar{\xi}_N(t))$. Summing this per-agent bound over v and applying Theorem 4's $\mathbb{E}[W_1(\bar{\xi}_N, \xi^\star)] \leq C(\mathsf{H})/\sqrt{N}$ lifts the compartment convergence to $\mathbb{E}|\bar{I}_N(t) - I(t)| \leq \rho_{\max} C(\mathsf{H})/\sqrt{N}$. Analogous bounds hold on $\bar{E}_N$, $\bar{S}_N$, $\bar{R}_N$ at the same rate. In the $N \to \infty$ limit, the empirical compartment fractions converge at $O(N^{-1/2})$ in expectation to the solution of the ODE system obtained by substituting the rates of Claims 1 through 3 into (27). Turnover at rate μ adds $-\mu E$ and $-\mu I$ terms by identical balancing. □

Corollary 12 (Per-Class SEIR for Block-Structured Topologies)**.** *Let agents partition into K homogeneity classes $V = V_1 \sqcup \cdots \sqcup V_K$ with within-class exchangeability, and let $\{\beta_{ij}\}_{i,j\in[K]}$ be the inter-class contact rates determined by the class-averaged typed-event intensity. Then the K-class SEIR ODE system*

$$\frac{dE_i}{dt} = S_i \sum_j \beta_{ij}\, I_j/|V_j| - \sigma_i E_i, \qquad \frac{dI_i}{dt} = \sigma_i E_i - \gamma_i I_i,$$

holds in the $N \to \infty$ limit, with the $\sqrt{N}\, W_1$ rate of Theorem 4 applied per class. For a hub-and-spoke topology ($K = 2$, hub class $|H| = O(1)$, spoke class $|S| = N - O(1)$), $\beta_{HS}, \beta_{SH} > 0$ and $\beta_{SS} \approx 0$, and the basic reproduction number is $R_0 = \sqrt{\beta_{HS}\beta_{SH}}/\gamma$, generalizing the scalar R_0 of Theorem 26. Proof via multi-type branching processes [53, 54] below.

Remark 13 (Non-exchangeable topologies)**.** *Hub-and-spoke and star topologies violate the exchangeability assumption of Theorem 4. Corollary 12 formalizes the remedy: the per-class SEIR limit holds with the same $\sqrt{N}\, W_1$ rate applied within each homogeneity class, and the cross-class coupling is absorbed into the K-type contact matrix (β_{ij}). The empirical slope-steepening reported in Figure 5 (hub-and-spoke vs. mesh) is consistent with the class-averaged rate predicted by Corollary 12. A full graphon mean-field extension remains future work.*

Proof of Corollary 12 (Per-class SEIR). Apply Theorem 4 restricted to each class V_i separately. Within-class exchangeability holds by construction, and (A2.5) restricts to a per-class constant $L_{\mathrm{drift},i}$ under the same bounded-coupling (A3). Cross-class coupling enters only through the inter-class contact rates β_{ij}, which by (A3) satisfy $\beta_{ij} \leq c_{\max,ij} \cdot \|B^{(r,\phi_i,\phi_j)}\|_{\mathrm{op}}$ on the aggregation axis. This is the multi-type branching-process structure of Ball et al. [53] (see also Britton [54]), with the SEIR threshold map specialized as in Corollary 5. For the star topology with $K = 2$, the hub class H has $|H| = O(1)$ (in Eve's running example $|H| = 1$) and degree $\Theta(N)$, while the spoke class S has $|S| = N - |H|$ and degree 1. The per-class empirical measures converge at $N^{-1/2}$ on the spoke class (classical rate) and trivially on the hub class. The cross-class contact matrix is $\begin{pmatrix} 0 & \beta_{HS} \\ \beta_{SH} & 0 \end{pmatrix}$, rank-1 under star connectivity. The basic reproduction number is $R_0 = \sqrt{\beta_{HS}\beta_{SH}}/\gamma$, and the mean-field rate on each class follows from Theorem 4. General K-class topologies are handled analogously with

the K-type matrix (β_{ij}); conditions for subcriticality (all eigenvalues below 1) generalize the scalar $R_0 < 1$ condition of Theorem 26. □

Two consequences for the main text. The attacker's choice of per-contact influence magnitude (the scale of $B^{(r,\cdot,\cdot)}$ entries) tunes σ^{-1} directly; this is the formal content of Remark 6. A defender that monitors the full latent belief b_t (Swarm Shepherd) rather than thresholded observations has access to the Exposed compartment before agents cross τ_I, which is the empirical basis of A1 (Section 5.2).

A.3 Proof of Theorem 1 (Swarm Shepherd Belief Consistency)

The proof composes three lemmas. Lemma 14 is a single-node Kalman sample-complexity bound adapted from [55]. Lemma 15 is the factored-belief error contraction of [38]. Lemma 16 is a hop union bound.

Lemma 14 (Single-node Kalman parameter-to-posterior error)**.** *Consider the reduced single-node Kalman filter at v with self-dynamics $A^{\phi(v)}$ and anchored emission H. Under assumptions (A2), (A4), and the stable self-dynamics part of (A7), the stationary Kalman gain is uniquely defined and locally Lipschitz in (A, H, Q, R) on a neighborhood of the true parameters with Lipschitz constant L_K. Building on the system-identification rates of [56, 57], Theorem 3.2 of Tsiamis et al. [55] gives, after n anchored observations, parameter error $\|(\hat{A}, \hat{H}, \hat{Q}, \hat{R}) - (A, H, Q, R)\|_F \le \epsilon_{\mathrm{par}}$ with probability at least $1 - \delta_1$ whenever $n \ge C_1\, d^2 \log(d/\delta_1)/(\epsilon_{\mathrm{par}}^2\, \lambda_{\min}^2\, \nu_{\min}^2)$, where $\nu_{\min}$ is the minimum singular value of the observability matrix $\mathcal{O}(A, H)$ and $\lambda_{\min}$ the minimum incident-event rate. Combining with Kalman-gain Lipschitzness and the Pinsker-type bound $\|b - b^\star\|_{\mathrm{TV}} \le C_G(\|\hat{\mu} - \mu^\star\|_2 + \|\hat{\Sigma} - \Sigma^\star\|_F)$ for Gaussian posteriors and setting $\epsilon_{\mathrm{par}} = \epsilon/(2L_K C_G)$ yields $\Pr[\|b^{\mathrm{single}}_{v,n} - b^\star_{v,n}\|_{\mathrm{TV}} \le \epsilon/2] \ge 1 - \delta_1$.*

Lemma 15 (Factored-belief approximation error)**.** *Boyen and Koller [38] show that for a factored belief $b^{\mathrm{fact}}_{v,t} = \prod_v b_{v,t}$ tracking a full-joint DBN, $\|b^{\mathrm{fact}}_{v,t} - b^{\mathrm{full}}_{v,t}\|_{\mathrm{TV}} \le \gamma^t\, \|b^{\mathrm{fact}}_{v,0} - b^{\mathrm{full}}_{v,0}\|_{\mathrm{TV}} + C_{\mathrm{fact}}\, L^{2k}$, with $\gamma < 1$ the DBN mixing contraction and L the bound on cross-level coupling (A3). The first term decays with t for stationary streams; the second dominates asymptotically and is the L^{2k} factor in the theorem.*

Lemma 16 (Hop union bound)**.** *Belief at v depends on events at nodes within k hops. There are at most $(d_{\max})^k$ such nodes, where $d_{\max}$ is the maximum influence-graph degree. Union-bounding over these nodes' Lemma 14 failure events (each with probability $\le \delta_1$) and absorbing the $(d_{\max})^k$ factor into the logarithmic dependence of n_0 yields total failure probability $\le \delta$.*

Proof of Theorem 1. Apply Lemma 14 to every node in the k-hop neighborhood of v. Lift through Lemma 15 to bound the factored-belief approximation by $\epsilon/2$. Apply Lemma 16 to union-bound the failure event. This gives $\Pr[\|b_{v,n} - b^\star_{v,n}\|_{\mathrm{TV}} \le \epsilon] \ge 1 - \delta$ whenever $n \ge n_0 = O(d^2\, L^{2k}\, \log(d/\delta)\, /\, \epsilon^2 \lambda_{\min}^2\, \nu_{\min}^2)$, which is Theorem 1. □

Lemma 17 (Short-horizon first-passage bound)**.** *Let X_t denote the scalar process after the true early-threshold crossing, with $X_0 = \tau_E$, $\tau_{\mathrm{FP}} = \inf\{t : X_t \ge \tau_I\}$, and $\Delta_\tau = \tau_I - \tau_E > 0$. Suppose that on the warning band $[\tau_E, \tau_I]$ the semimartingale decomposition is $dX_t = a_t\, dt + \sigma_t\, dW_t$ with $|a_t| \le \bar{a}$ and $\sigma_t^2 \le \bar{\sigma}^2$. For any horizon $h > 0$ with $\Delta_\tau > \bar{a}h$,*

$$\Pr[\tau_{\mathrm{FP}} \le h] \le \min\left\{1,\, 2\exp\left(-\frac{(\Delta_\tau - \bar{a}h)^2}{2\bar{\sigma}^2 h}\right)\right\}. \tag{19}$$

Proof. Write $X_t = \tau_E + \int_0^t a_s\, ds + M_t$, where $M_t = \int_0^t \sigma_s\, dW_s$ and $\langle M \rangle_t \le \bar{\sigma}^2 t$. If $\tau_{\mathrm{FP}} \le h$, then $\sup_{0 \le t \le h} M_t \ge \Delta_\tau - \bar{a}h$. By the Dambis–Dubins–Schwarz time change and the Brownian reflection principle,

$$\Pr\left[\sup_{0 \le t \le h} M_t \ge a\right] \le \Pr\left[\sup_{0 \le u \le \bar{\sigma}^2 h} B_u \ge a\right] = 2\{1 - \Phi(a/\sqrt{\bar{\sigma}^2 h})\} \le 2\exp\left(-\frac{a^2}{2\bar{\sigma}^2 h}\right)$$

with $a = \Delta_\tau - \bar{a}h > 0$. See Borodin and Salminen [58] Chapter I for Brownian first-passage formulae and Jacod and Shiryaev [52] for the martingale time-change machinery. □

Proof of Corollary 2. By Theorem 1, Swarm Shepherd's filtered posterior tracks the oracle posterior to within ϵ TV with probability $\geq 1-\delta$. On this event, the warning-time localization condition gives $\hat{T}_v \leq T^\star_{E,v} + \eta_\epsilon$. If the true first-passage time from τ_E to τ_I exceeds $k + \eta_\epsilon$, then $\hat{T}_v + k \leq \tau^\star_v$. Lemma 17 with $h = k + \eta_\epsilon$ bounds the complementary event by the exponential term in (13). Combining the two failure events via union bound gives $\Pr[\hat{T}_v + k \leq \tau^\star_v]$ bounded as in (13). □

Eve's numbers. For Eve's ecosystem the $k = 3$ certificate is evaluated from the A1 calibration estimates of the threshold margin, warning-band drift, diffusion, and localization slack:

$$\epsilon_{\text{lead}}(3) = \min\left\{1,\ 2\exp\left(-\frac{(\widehat{\Delta}_\tau - \widehat{a}(3+\widehat{\eta}_\epsilon))^2}{2\widehat{\sigma}^2(3+\widehat{\eta}_\epsilon)}\right)\right\}.$$

The A1 warning-lead result reports the empirical conformal coverage of this $k = 3$ certificate (Section 5.2, Table 14); the theorem supplies the sufficient condition, while the experiment verifies that the calibrated deployment satisfies the target.

A.4 Proof of Theorem 7 (Two-Scale Deployment Regime)

We first isolate two auxiliary lemmas used by the proof, then assemble the bound through a triangle decomposition of the end-to-end error.

Lemma 18 (Bounded-density continuity modulus)**.** *Suppose the marginal density of a random variable ξ on $\mathbb{R}$ is uniformly bounded by $\rho_{\max}$ on a neighborhood of τ_I. Then for any random variables X, Y on $\mathbb{R}$, $\mathbb{E}\big|\mathbb{1}[X \geq \tau_I] - \mathbb{1}[Y \geq \tau_I]\big| \leq \rho_{\max} W_1(X, Y)$, and for any scalar estimator $\hat{\xi}$ of ξ, $\Pr\big[\mathbb{1}[\hat{\xi} \geq \tau_I] \neq \mathbb{1}[\xi \geq \tau_I]\big] \leq \rho_{\max}\,\mathbb{E}|\hat{\xi} - \xi|$.*

Proof. A standard bounded-density argument; see Tsybakov [59, Theorem 6.1]. The indicator disagreement set is contained in $\{|\xi - \tau_I| \leq |X - Y|\}$, whose probability under bounded density $\rho_{\max}$ near τ_I is at most $\rho_{\max}\,\mathbb{E}|X - Y|$; the Kantorovich dual gives the W_1 form. □

Lemma 19 (Density inheritance under exchangeable averaging)**.** *Let $X_1, \ldots, X_N$ be exchangeable random variables on $\mathbb{R}$ with each marginal density bounded by $\rho_{\max}$ on a neighborhood U of a threshold τ. Then the density of the empirical mean $N^{-1}\sum_v X_v$ is bounded by $\rho_{\max}$ on U.*

Proof. For independent X_v, the density of the mean is an N-fold convolution $f_{\bar{X}} = f_1 * \cdots * f_N$ on scaled variables. Young's inequality gives $\|f_{\bar{X}}\|_\infty \leq \min_v \|f_v\|_\infty \leq \rho_{\max}$ on U. Exchangeable (possibly dependent) X_v preserve the bound. The joint density marginalizes to the empirical-mean density through the same convolution structure after conditioning on the exchangeable de Finetti mixing measure, and the L^∞ bound is preserved under mixtures. Hence $\|f_{\bar{X}}\|_\infty \leq \rho_{\max}$ on U. □

Lemma 20 (TV-to-scalar for Gaussian marginals)**.** *Let $b, b^\star$ be d-dimensional Gaussian posteriors on a latent state z with $\|b - b^\star\|_{\text{TV}} \leq \epsilon$, and let $\xi = e^\top z$ be a fixed linear aggregation with $\|e\| = 1$. Then $\mathbb{E}|\xi_b - \xi_{b^\star}| \leq C_{\text{Pinsker}}\,\sigma_{\max}\sqrt{\epsilon}$, where $\sigma^2_{\max}$ bounds the posterior variance along e under both b and $b^\star$.*

Proof. By Pinsker's inequality $\|b - b^\star\|_{\text{TV}} \leq \sqrt{\text{KL}(b\,\|\,b^\star)/2}$. The KL between two Gaussians is a quadratic form in mean and covariance differences, and the marginal along e is Gaussian with standard deviation at most $\sigma_{\max}$. Combining gives the stated bound. □

Proof of Theorem 7. We prove the bound for the canonical SEIR Infected functional $\hat{M}_N(t) = \hat{I}_N(t)$ guaranteed by Corollary 5. For a general 1-Lipschitz functional of the latent aggregate the bound holds directly by Kantorovich–Rubinstein duality without the bounded-density assumption; for the threshold indicator used here the indicator is not Lipschitz, and Lemma 18 substitutes for Lipschitzness.

Step A (triangle decomposition). Let $\bar{I}_N(t) = N^{-1}\sum_v \mathbb{1}[\xi_{v,t} \geq \tau_I]$ denote the oracle compartment fraction (same population as $\hat{I}_N$ but using the true latent trajectory $\xi_{v,t}$) and let $I(t)$ denote the SEIR

ODE solution. The triangle inequality gives

$$\left|\hat{I}_N(t) - I(t)\right| \;\le\; \underbrace{\left|\hat{I}_N(t) - \bar{I}_N(t)\right|}_{\text{(i) filter aggregation}} + \underbrace{\left|\bar{I}_N(t) - I(t)\right|}_{\text{(ii) mean-field}}.$$

Step B (mean-field term). The bounded-density assumption of Theorem 7 bounds the per-agent marginal density of $\xi_{v,t}$ at τ_I by $\rho_{\max}$. By Lemma 19, the aggregated marginal density of $\bar{\xi}_N(t) = N^{-1}\sum_v \xi_{v,t}$ is also bounded by $\rho_{\max}$ on the same neighborhood of τ_I. Theorem 4 gives $\mathbb{E}[W_1(\bar{\xi}_N(\cdot), \xi^\star(\cdot))] \le C(\mathsf{H})/\sqrt{N}$. Apply Lemma 18 with $X = \bar{\xi}_N(t)$ and $Y = \xi^\star(t)$ at the threshold τ_I: $\mathbb{E}\left|\bar{I}_N(t) - I(t)\right| \le \rho_{\max} \cdot \mathbb{E}[W_1(\bar{\xi}_N(t), \xi^\star(t))] \le C_2(\mathsf{H})/\sqrt{N}$ with $C_2(\mathsf{H}) = \rho_{\max}\, C(\mathsf{H})$.

Step C (filter term). Theorem 1 guarantees $\|b_{v,n} - b^\star_{v,n}\|_{\mathrm{TV}} \le \epsilon$ with probability $\ge 1-\delta$ whenever $n \ge n_0$ in (12). Apply Lemma 20 at each node v with the aggregation vector e of Section 4.4. This yields the expected scalar error $\mathbb{E}|\hat{\xi}_{v,t} - \xi_{v,t}| \le C_{\mathrm{Pinsker}}\, \sigma_{\max}\, \sqrt{\epsilon}$. Conditioning Lemma 18 on $\hat{\xi}_{v,t}$ gives the pointwise bound $\Pr[\mathbb{1}[\hat{\xi}_{v,t} \ge \tau_I] \ne \mathbb{1}[\xi_{v,t} \ge \tau_I] \mid \hat{\xi}_{v,t}] \le \rho_{\max}\, |\hat{\xi}_{v,t} - \xi_{v,t}|$. Taking expectation over $\hat{\xi}_{v,t}$ on both sides (linearly, since the right-hand side is linear in the random variable) gives the unconditional bound $\Pr[\text{disagreement at node } v] \le \rho_{\max}\, \mathbb{E}|\hat{\xi}_{v,t} - \xi_{v,t}|$. Averaging the indicator-disagreement probabilities over the N agents and propagating through the k-hop factored-belief contraction (Lemma 15) composed with the hop union bound (Lemma 16) injects the L^k factor by the same argument used to prove Theorem 1. The result is

$$\mathbb{E}\left|\hat{I}_N(t) - \bar{I}_N(t)\right| \;\le\; \rho_{\max} \cdot \frac{C_1\, L^k}{\nu_{\min}\sqrt{n}},$$

with C_1 absorbing $C_{\mathrm{Pinsker}}\, \sigma_{\max}$ and the Kalman sample-complexity constants of Lemma 14.

Step D (composition). Combining Steps B and C yields the stated expectation bound. Setting the two expected-error terms equal, $\rho_{\max}\, C_1\, L^k/(\nu_{\min}\sqrt{n}) = C_2(\mathsf{H})/\sqrt{N}$, and solving for n identifies the *balance curve* $n_\circ(N)$ in (17). The theorem's claim is in expectation. $n_\circ(N)$ is the sample complexity at which the expected filter contribution equals the expected mean-field contribution. For individual trajectories, the two error contributions concentrate around their expectations by Markov's inequality applied to the non-negative error bounds in Steps B and C. The balance point is therefore robust to bulk-trajectory fluctuations. Below $n_\circ(N)$ the filter term dominates in expectation, so the marginal return on n is higher. Above it the mean-field term dominates, so the marginal return on N is higher.

The scaling relation $n_\circ(N) \propto N$ is the load-bearing content of the theorem. The proportionality constant depends on $(L, k, \nu_{\min}, \rho_{\max}, C_1, C_2(\mathsf{H}))$. A high-probability statement replaces these central constants with worst-case upper bounds valid over a parameter class. The resulting boundary is parallel to $n_\circ(N)$ but shifted upward, and the scaling law is unchanged. The value 8.3 reported in Section 4 uses EM central estimates from held-out benign traces. It is therefore an operating-point value at the expectation balance curve, not a worst-case constant. □

Relation to two-timescale stochastic approximation. The decomposition $E_{\mathrm{filter}}(n) + E_{\mathrm{MF}}(N)$ is structurally analogous to the two-timescale error decompositions in [60, 61]. A fast scale (inference convergence in n) composes with a slow scale (population aggregation in N). The optimal deployment regime is determined by balancing the two rates. Compared to those RL and stochastic-approximation settings, our slow scale is a propagation-of-chaos rate rather than a step-size decay.

Relation to Bayesian McKean–Vlasov inference. Ning and Wu [62] study Bayesian inference for partially observed McKean–Vlasov SDEs but do not compose the resulting filter rate with a mean-field rate to give a deployment regime. Theorem 7 is the specialization of their inference view to heterogeneous event graphs with anchored emissions, and its composition with Theorem 4 is new.

A.5 Proof of Theorem 27 (Optimal Activation Time)

Proof. The adversary maximizes $J(T) = e^{-qT}(1 - 1/R_0(T))$ where $R_0(T) = \kappa\, r(T)$ with $\kappa = c_{\max}\alpha/\gamma$, and $r(T) = r_{\max} - (r_{\max} - r_0)e^{-\lambda T}$. Then $r'(T) = \lambda(r_{\max} - r(T))$ and $R_0'(T) = \kappa\, r'(T)$.

Setting $dJ/dT = 0$ and applying the product rule gives

$$-q\,e^{-qT}\left(1-\frac{1}{R_0}\right)+e^{-qT}\frac{R_0'}{R_0^2}=0,$$

so $R_0'(T^\star) = q\,R_0(T^\star)(R_0(T^\star)-1)$. Substituting $R_0 = \kappa r$ gives the stationarity condition $\lambda(r_{\max}-r^\star) = q\,r^\star(\kappa r^\star - 1)$ for $r^\star = r(T^\star)$, that is,

$$q\kappa(r^\star)^2+(\lambda-q)\,r^\star-\lambda r_{\max}=0, \qquad r^\star=\frac{-(\lambda-q)+\sqrt{(\lambda-q)^2+4q\kappa\lambda r_{\max}}}{2q\kappa},$$

where $r^\star$ is the feasible (positive) root. Inverting $r(T)$ yields

$$T^\star=\frac{1}{\lambda}\ln\frac{r_{\max}-r_0}{r_{\max}-r^\star},$$

which is interior whenever $r_0 < r^\star < r_{\max}$. Because the left-hand side of the stationarity condition is positive for $r^\star < r_{\max}$, any interior root satisfies $\kappa r^\star > 1$, so $R_0(T^\star) > 1$.

Second-order condition. Write $dJ/dT = \kappa e^{-qT}h(r(T))/R_0(T)^2$ with $h(r) = \lambda(r_{\max}-r) - q\,r(\kappa r-1)$. Since $h'(r) = -\lambda - q(2\kappa r-1) < 0$ whenever $\kappa r > 1/2$, and $r(T)$ is strictly increasing, dJ/dT changes sign from positive to negative at $T^\star$. The condition $\kappa r^\star > 1$ enforced by stationarity therefore makes $T^\star$ a strict maximum.

Monotonicity. At the feasible root, $T^\star$ decreases in both λ and q: faster trust accumulation reaches the profitable routing share sooner, and a higher removal hazard favors earlier activation (Table 4).

The submitted version displayed a closed form for $T^\star$ that omitted κ and therefore did not solve its own stationarity condition; the expression above replaces it, and Table 4 is regenerated from it. No defense result consumes $T^\star$. □

Joint T-σ optimization. The adversary jointly chooses Phase 0 duration T and post-activation incubation rate σ. T controls pre-activation trust accumulation, affecting β through routing share $r(T)$. σ controls post-activation stealth, affecting incubation duration $1/\sigma$. The two parameters are independent: T is chosen before activation, σ is chosen after. The joint objective is $J(T,\sigma) = e^{-qT}\cdot(1-1/R_0(T,\sigma))$ where $R_0(T,\sigma) = c_{\max}\cdot r(T)\cdot\alpha\cdot\sigma/((\sigma+\mu)(\gamma+\mu))$. The separable structure means $T^\star$ depends only on $(\lambda, q, \kappa, r_0, r_{\max})$ and σ^* depends only on (γ,μ) and the adversary's detection-avoidance constraint.

Sensitivity of T^*. Table 4 shows T^* for representative parameter combinations.

Table 4: Optimal activation time $T^\star$ (sessions) for varying gateway learning rate λ and removal hazard q, computed from the feasible root of Theorem 27.

	$q = 0.001$	$q = 0.005$	$q = 0.01$	$q = 0.05$	$q = 0.1$
$\lambda = 0.01$	353.04	230.09	192.95	145.80	137.46
$\lambda = 0.05$	100.76	70.61	**58.98**	38.59	33.32
$\lambda = 0.1$	57.18	41.61	35.30	23.01	19.30
$\lambda = 0.5$	14.63	11.44	10.08	7.06	5.90

Computed with $r_0 = 0.2$ ($K = 5$ tools), $r_{\max} = 0.5$, and $\kappa = 2.4$ from the logged Eve parameters. Bold marks Eve's reference cell. $T^\star$ decreases in both λ and q; the $\lambda = 0.5$, $q = 0.1$ cell is an exact tenfold rescaling of the reference cell.

Model limitations. The optimal activation model assumes: (1) gateway routing follows exponential saturation, which may not hold for all routing algorithms; (2) the hazard rate q is constant, whereas suspicious behavior could increase q over time; (3) R_0 depends only on routing share, whereas memory store density (Proposition 23) also contributes to β. These simplifications enable a closed-form solution at the cost of excluding adaptive defender responses.

A.6 SEIR Derivations and Stability Analysis

A.6.1 Disease-Free Equilibrium

The disease-free equilibrium (DFE) of the SEIR system (27) is $(S^*, E^*, I^*, R^*) = (N, 0, 0, 0)$. Including agent turnover at rate μ (agents created and retired), the infected subsystem becomes $dE/dt = \beta I - (\sigma + \mu)E$ and $dI/dt = \sigma E - (\gamma + \mu)I$ at the DFE. The Jacobian is

$$J = \begin{pmatrix} -(\sigma + \mu) & \beta \\ \sigma & -(\gamma + \mu) \end{pmatrix}.$$

The DFE is unstable (infection grows) when the dominant eigenvalue is positive. This occurs when $\beta\sigma > (\sigma + \mu)(\gamma + \mu)$, i.e., $R_0 = \frac{\beta\sigma}{(\sigma+\mu)(\gamma+\mu)} > 1$.

A.6.2 R0 Derivation via Next-Generation Matrix

Using the next-generation matrix method, the transmission matrix F and transition matrix V for the infected compartments (E, I) are

$$F = \begin{pmatrix} 0 & \beta \\ 0 & 0 \end{pmatrix}, \quad V = \begin{pmatrix} \sigma + \mu & 0 \\ -\sigma & \gamma + \mu \end{pmatrix}.$$

The next-generation matrix is FV^{-1}, and $R_0 = \rho(FV^{-1}) = \frac{\beta\sigma}{(\sigma+\mu)(\gamma+\mu)}$. When $\mu = 0$ (no agent turnover), this simplifies to $R_0 = \frac{\beta\sigma}{\sigma\gamma} = \frac{\beta}{\gamma}$, recovering the standard SIR threshold.

A.6.3 Herd Immunity Threshold

If a fraction p of agents are immune (in compartment R), the effective reproduction number is $R_{\text{eff}} = R_0(1 - p)$. Setting $R_{\text{eff}} = 1$ yields the herd immunity threshold $H = 1 - 1/R_0$.

A.7 Amplification Results

Propositions 22 and 23 characterize two secondary amplification effects in agentic ecosystems beyond the base SEIR prediction. Full statements and proofs are below; the main-text summary appears at the end of Section 4.4. Corollary 21 composes Theorems 4 and 7 with a forward-cone argument to bound the downstream impact of a localized entry identified by Proposition 3.

Corollary 21 (Impact Propagation from a Localized Entry)**.** *Given an entry (u, ℓ, t_0) identified by the trace-back of Proposition 3, the expected fraction of compromised agents at horizon $t_0 + \Delta$ along the forward cone of (u, t_0) is bounded by*

$$\mathbb{E}\Big[\, \hat{M}_N(t_0 + \Delta) \,\Big] \;\le\; \rho_{\max}\, e^{(\beta-\gamma)\,\Delta} + \frac{C(\Delta)}{\sqrt{N}}, \tag{20}$$

with β, γ the SEIR rates of Corollary 5 and $C(\Delta)$ the horizon-dependent constant of Theorem 4. A defender with budget B on intervention magnitude can choose to act at any Δ for which the right-hand side exceeds an acceptable spread.

Proposition 22 (Cycle Amplification)**.** *In a contact graph with cycles of length k, let $\delta > 0$ be the per-cycle drift increment (an eigenvalue of $B^{(\mathsf{handoff},\mathcal{A},\mathcal{A})}$ along the aggregation axis) and d_0 the initial drift. After j completed cycles the drift satisfies $d_j \ge d_0(1 + \delta)^j$, and after m directed edge transitions on a k-cycle it satisfies $d(m) \ge d_0(1 + \delta)^{\lfloor m/k \rfloor}$. This yields exponential amplification in frameworks with graph cycles (e.g., LangGraph).*

Proof. Consider a cycle of length k in the contact graph. Let $d_0 > 0$ be the initial drift magnitude at a node in the cycle. Let $\delta > 0$ be the per-iteration drift increment, the multiplicative factor by which drift increases when a node processes drifted input from its predecessor. After one complete cycle (k steps), the drift returns to the originating node with magnitude $d_0(1 + \delta)$. After j completed cycles (jk edge transitions), the drift at the originating node satisfies $d_j = d_0(1 + \delta)^j$. Since each cycle requires k transitions, after m edge transitions at least $\lfloor m/k \rfloor$ cycles have completed, so $d(m) \ge d_0(1 + \delta)^{\lfloor m/k \rfloor}$. This is exponential in m with base $(1 + \delta)^{1/k}$, confirming that graph

cycles produce exponential drift amplification. For LangGraph with $\delta = 0.02$ and $k = 2$ (minimal cycle), $m = 30$ edge transitions give $d(30) \geq 1.02^{15} d_0 \approx 1.346\, d_0$, a 34.6% cumulative shift, and 30 completed cycles ($m = 60$) give $1.02^{30} d_0 \approx 1.811\, d_0$, an 81.1% shift. Statement, proof, and the measurements of Appendix D.8 index drift by edge transitions throughout. □

Proposition 23 (Density Amplification)**.** *When n_A agents are compromised via Case A and write to a shared memory store, the effective retrieval probability for poisoned entries increases as $p \approx 1 - (1 - p_{base})^{n_A}$. This raises the effective β for Case C agents through the spectral radius of Theorem 4.*

Proof. Let p_{base} be the probability that a single poisoned entry is retrieved in a given query. When n_A compromised agents independently write poisoned entries to a shared memory store, the probability that *at least one* poisoned entry is retrieved is $p = 1-(1-p_{\text{base}})^{n_A}$. For $p_{\text{base}} = 0.479$ (MemoryGraft retrieval proportion) and $n_A = 3$, we get $p = 1 - 0.521^3 \approx 0.859$. This raises the effective infection rate β for Case C agents, as the transmission probability per contact increases with poisoned entry density. □

A.8 Proposition: Calibration Consistency (Statement and Proof)

The statement of Proposition 24 (referenced from §4.5 and §6) is reproduced here for locality with its proof.

Proposition 24 (Calibration Consistency)**.** *Under joint observability (Lemma 25) and a persistence-of-excitation condition on the typed-event stream (assumption (A6), Appendix B.7), expectation–maximization on M benign windows each of length H attains*

$$\|\hat{\theta} - \theta\|_2 \;=\; O_P\!\left(\sqrt{\tfrac{d^2}{M\mathsf{H}}}\right) \tag{21}$$

under benign stationarity. If the deployment distribution differs from calibration at total variation ϵ, an additive $\epsilon^{1/2}$ term enters (21)*.*

Proof. Let $\theta = (A^\phi, B^{(r,\cdot,\cdot)}, H, Q^\phi, \Sigma_{\text{obs}})$ be the parameter vector of the factored state-space model of Definition 4 (the intervention matrix C is set from deterministic action semantics and not included in θ), and let $\hat{\theta}$ be the EM maximiser of the joint log-likelihood on M benign windows of length H.

Identifiability. Under joint observability (Lemma 25), the pair (A^ϕ, H) is observable on the 6-dimensional state with minimum singular value $\nu_{\min} > 0$ of the k-step observability Gramian. Under the persistence-of-excitation assumption (A6) on the typed-event stream (Appendix B.7), the sample covariance of event-induced innovations concentrates at rate $O_P(1/\sqrt{M\mathsf{H}})$ to a strictly positive-definite limit. The Fisher information $\mathcal{I}(\theta)$ is therefore non-singular at the truth, which is the standard identifiability condition for linear-Gaussian state-space models [55].

Consistency rate. Standard M-estimation on the quasi-log-likelihood of the factored model (single-node analysis in [55], extended via the factored-belief contraction of [38]) yields

$$\sqrt{M\mathsf{H}}\,(\hat{\theta} - \theta) \;\xrightarrow{d}\; \mathcal{N}\big(0,\, \mathcal{I}(\theta)^{-1}\big)\,,$$

which gives $\|\hat{\theta} - \theta\|_2 = O_P(\sqrt{d^2/(M\mathsf{H})})$ by bounding $\|\mathcal{I}(\theta)^{-1}\|_{\text{op}} \leq c/\nu_{\min}^2$ and noting that the parameter dimension is $O(d^2)$ under the sparsity of $B^{(r,\cdot,\cdot)}$ induced by the edge-type partition.

Distribution shift. Suppose the deployment distribution P^{dep} differs from the calibration distribution P^{cal} at $d_{\text{TV}}(P^{\text{dep}}, P^{\text{cal}}) \leq \epsilon$. By the Pinsker/TV transfer bound and the quadratic smoothness of the log-likelihood at θ, the population-level objectives satisfy $\|\theta^{\text{dep}} - \theta^{\text{cal}}\|_2 \leq C_\ell \epsilon^{1/2}$ for a universal C_ℓ depending on the likelihood curvature. The triangle inequality gives the additive $\epsilon^{1/2}$ term of (21). □

B Formal Model Details

This appendix carries extended formal detail beyond the main text. It enumerates the observable edge types and shared latent coordinates, gives the full factored-DBN factorization, states the defender POMDP and attacker constrained MDP in their full form, documents the identifiability argument, describes the conformal calibration protocol, reproduces the assumptions-to-results map, and hosts the support tables and figures that Sections 1–4 reference.

B.1 Main-Text Maps and Positioning

The main paper keeps the story on the **WHEN / WHICH / HOW** spine. The roadmap table, related-work comparison, and dual-view schematic are reproduced here to preserve technical traceability without consuming main-text pages.

Positioning. Our closest conceptual peer is Motwani et al. [63], which studies steganographic communication among colluding agents. Where that work models covert channels, we study persistent latent compromise: a legitimate-looking agent accumulates trust, leaves cross-layer residue, and must be detected, traced, and contained from partial runtime evidence. Swarm Shepherd fills this gap by treating runtime traces as evidence of hidden compromise state and connecting early warning, layer-level trace-back, and containment within a single inference framework. Theory-facing comparisons to Bayesian McKean–Vlasov inference [62] and two-timescale stochastic approximation [60, 61] are addressed by the proof structure in Appendix A.

Table 5: Pillar-depth summary. Each pillar contributes both a formal guarantee and a paired experiment.

Pillar	Formal guarantee	Empirical validation
Two-scale regime **(headline)**	Theorem 7	A2 phase diagram (Fig. 6)
Swarm Shepherd inference	Theorem 1; Corollary 2	A1 $\sqrt{n}$ rate fit (§5.2)
Mean-field theory	Theorem 4; Corollary 12	A3 $\sqrt{N}$ fit (§5.4)
Trace-back smoother	Proposition 3	A4: $24 \times 3 \times 30$ seeds
Containment	Theorem 26; Corollary 5	Live R_c (Table 3); forecast (Table 27)
Threat model	Definition 4; 24-variant taxonomy	§5 ASR

Table 6: Comparison with representative prior work. Among the compared lines of work, the Agentic Sleeper is the only setting that combines trust-building activation, SEIR epidemic propagation with adversary-controlled incubation, multi-level persistence, cross-level latent state, and factored-POMDP defense.

Method	Activation	Propagation	Incubation	Persistence	Formal Model	Latent State	Control
Prompt Injection	Immediate	None	None	None	None	N	None
Temporal Attacks	Discrete trigger	None	Fixed	Layer	Taxonomy	N	None
Agent Smith	Immediate	$O(\log N)$	None	Session	SIS-like	N	None
Morris II	Immediate	Worm	None	Code	None	N	None
MemoryGraft	Immediate	None	None	Memory	None	N	None
HEARTBEAT	Passive	None	None	Memory	E→M→B	N	None
DeepContext [16]	N/A (defense)	None	None	Session	Stateful classifier	Intent drift	Per-turn classifier
Ours	**Trust → gradual**	**SEIR epidemic**	**Adversary-ctrl.**	**Multi-level**	**Factored POMDP**	**Y (cross-level)**	**Factored POMDP**

B.2 Observable Edge Types

The observable event graph $\mathcal{G}_t^{\text{obs}}$ of Definition 2 carries eleven typed edges that together span the three attack levels. Memory activity covers read, write, and retrieve. Logic activity covers invoke, return, modify, and handoff. Infrastructure activity covers grant, install, execute, and egress.

Memory edges (L1). A read edge fires when an agent pulls a fixed file or database row. A write edge fires when an agent persists a new memory entry. A retrieve edge fires when a vector- or semantic-search call returns a ranked result; the retrieval probability distribution determines the Case-C trigger rate $\alpha = f(p_{\text{retrieval}})$.

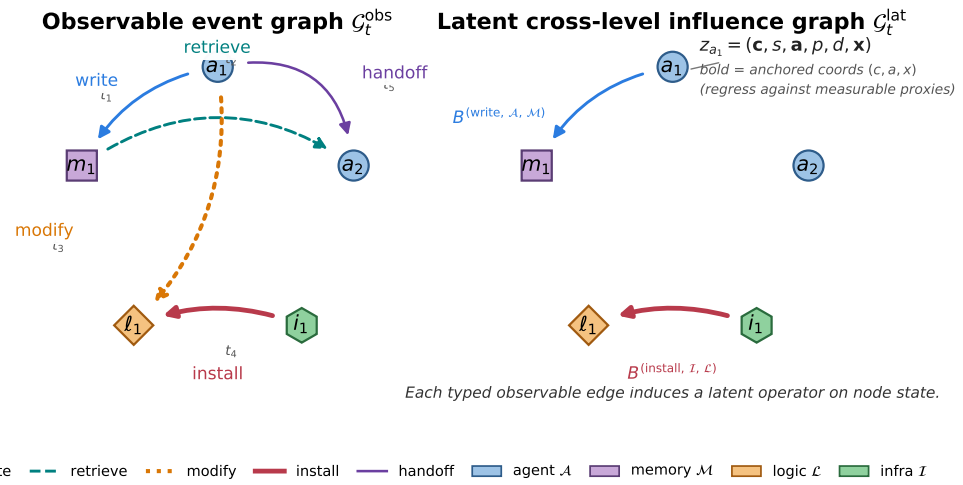

Figure 3: Dual-view event graph. Left: the observable event graph $\mathcal{G}_t^{\text{obs}}$ records typed, timestamped edges between agents, memory, logic, and infrastructure nodes. Right: the latent influence graph $\mathcal{G}_t^{\text{lat}}$ shares the node set, with a 6-dimensional latent state $z_v = (c, s, a, p, d, x)$ per node; the anchored coordinates (c, a, x) are regressed against measurable proxies. Each typed observable edge induces a typed influence operator $B^{(r,\phi(u),\phi(v))}$ that updates latent state. Swarm Shepherd (§4.2) filters this joint object from the event stream.

Logic edges (L2). An invoke edge fires on any tool or subagent call, return marks the reply, modify fires whenever a configuration, system prompt, tool schema, or routing rule is rewritten on disk or in memory, and handoff fires on agent-to-agent delegations that transfer control without a full tool invocation.

Infrastructure edges (L3). A grant edge fires whenever a permission or credential scope is extended. An install edge fires on package or binary installation, including `npx -y` auto-updates. An execute edge fires on a new process start attributable to an agent. An egress edge fires on any external network call initiated by the agent or its tools.

These eleven types are the minimum set that covers every attack variant in Table 10, 11, and 12 of Appendix C.8; each variant names the edge type its mechanism emits first.

B.3 Shared Latent Coordinates

Each node $v \in V^{\text{lat}}$ carries the six shared coordinates $z_{v,t} = (c, s, a, p, d, x)$ of Definition 3. The verbal interpretations below are calibrated against the anchored proxies (Appendix B.7) and checked against the controlled-injection traces of Appendix D.

Contamination $c_{v,t}$ (anchored). How much poisoned, unverified, or adversarial content the node has absorbed through retrieve, read, or handoff edges. Anchored to the poisoned-entry retrieval rate on a rolling window.

Susceptibility $s_{v,t}$ (unanchored). How readily the node accepts new cross-level influence; rises with prior contamination and with the attacker's Phase-0 trust accumulation. Estimated through its influence on the transition law.

Authority $a_{v,t}$ (anchored). How much routing weight, tool-selection priority, or delegation trust the node commands. Anchored to empirical routing share for agents and tools, or retrieval rank for memory.

Persistence $p_{v,t}$ (unanchored). How durably latent state survives restart, reindex, or configuration refresh. Rises with distributed-fragment persistence (Appendix C.5).

Deviation $d_{v,t}$ (unanchored). How far agent behavior has drifted from its baseline along the active drift primitive. Estimated through ASI subscales (Appendix D).

Capability exposure $x_{v,t}$ (anchored). How many sensitive resources the node can now reach. Anchored to the granted-permission count plus the reachable-resource count.

The three anchored coordinates (c, a, x) carry identifiability. The three unanchored coordinates (s, p, d) are constrained through the transition dynamics; estimation protocol is in Appendix B.7.

B.4 Type-Specific Latent Dimensions

Beyond the six shared coordinates, each node type carries additional coordinates summarized below. In the main text we use only the shared core so that results apply across node types; type-specific coordinates are used only in Appendix D ablations.

Agent nodes. Goal fidelity, tool preference, routing bias, plan brittleness, secrecy discipline, approval conservatism.

Memory nodes. Retrieval frequency, semantic centrality, provenance confidence, contamination density, recency and decay.

Logic nodes. Config integrity, prompt rigidity, schema corruption level, guard strength, approval-requirement strength.

Infrastructure nodes. Privilege level, sandbox-escape risk, deployment reach, restart persistence, audit visibility.

Each of these is a bounded scalar and enters the transition in Definition 4 as an additional coordinate of z_v; block-diagonal structure in $A^{\phi(v)}$ keeps type-specific coordinates local to their node class while the shared core is coupled across types via the typed influence operators $B^{(r,\phi(u),\phi(v))}$.

B.5 Factored DBN and POMDP Formalization

Transition factorization. With node-type self-dynamics $A^{\phi(v)}$ and typed influence operators $B^{(r,\phi(u),\phi(v))}$ the joint transition is

$$P(Z_{t+1} \mid Z_t, E_t^{\text{obs}}, U_t) \;=\; \prod_{v \in V^{\text{lat}}} P\big(z_{v,t+1} \mid z_{v,t}, \{z_{u,t}\}_{u \in N_{\text{in}}(v,t)}, \iota_{v,t}\big)\,, \tag{22}$$

where $N_{\text{in}}(v,t) = \{u : (u, v, r) \in E_t^{\text{obs}} \text{ for some } r\}$. This factorization is what makes the model a dynamic Bayesian network (DBN) whose per-step inference cost is linear in $|E_t^{\text{obs}}|$ rather than in $|V^{\text{lat}}|^2$.

Linear-Gaussian instantiation (v1 experiments). We instantiate Definition 4 with linear-Gaussian self-dynamics and typed influence operators. Given this structure, the per-node belief $b_{v,t}(z_v) = \mathcal{N}(\mu_{v,t}, \Sigma_{v,t})$ admits closed-form Kalman updates under the fixed event graph $E_{0:t}^{\text{obs}}$, with neighbor messages exchanged once per event. Parameters $(A^{\phi(v)}, B^{(r,\cdot,\cdot)}, H, Q^{\phi(v)}, \Sigma_{\text{obs}})$ are estimated by EM on benign calibration windows only, with no attack labels (Table 41); the intervention matrix C is calibrated from deterministic action semantics rather than learned by EM. Discrete-state, switching, and non-Gaussian alternatives are compared in Appendix E.5; neural-ODE extensions are left to future work.

Factored POMDP belief propagation. For the defender POMDP (Definition 5, below) the joint belief is intractable in closed form at scale, so we approximate it in fully factored form $b_t(Z) \approx \prod_v b_{v,t}(z_v)$ with neighbor-message updates analogous to loopy belief propagation. This is the standard factored-POMDP approach taken by [12, 14]; we inherit its convergence and approximation properties.

Attacker policy class. The attacker constrained MDP (Definition 6, below) admits a rich policy class; for Theorem 27 we restrict to threshold policies parameterized by activation time T and post-activation influence magnitude σ. More general policies are discussed in Appendix A.5 under "Joint T-σ optimization."

B.6 Decision Problems: Full Formalizations

Section 4.5 points to these coupled decision problems; full tuples are below.

Defender factored POMDP (full tuple).

Definition 5 (Defender Factored POMDP)**.** *The defender problem is* $(\mathcal{S}, \mathcal{U}_{\text{def}}, \mathcal{O}, P, \mathcal{Z}, \mathcal{R}_d, \gamma_d, b_0)$ *with hidden state* $\mathcal{S} = \prod_v \mathbb{R}^{d_v}$*, budgeted action space* $\mathcal{U}_{\text{def}}$ *(sanitize memory, freeze config, down-weight tool, isolate agent, require approval, no-op), observation space* $\mathcal{O}$ *containing the typed event stream and anchored emissions, transition kernel* P *and emission kernel* $\mathcal{Z}$ *from Definition 4, and risk-weighted reward* $\mathcal{R}_d$ *trading residual compromise against intervention cost. The belief* $b_t = P(Z_t \mid y_{0:t}, E^{\text{obs}}_{0:t})$ *is approximated in factored form* $b_t(Z) \approx \prod_v b_{v,t}(z_v)$ *with neighbor-message updates. The five defense paradigms of Appendix B.11 are instantiations of this controller under different action restrictions and belief summaries. Factorization matters because a per-node belief keeps inference tractable at deployment scale, and the anchored emission on* (c, a, x) *keeps each per-node belief identifiable rather than collapsing into a latent degeneracy.*

Attacker constrained MDP (full utility decomposition).

Definition 6 (Attacker Constrained MDP)**.** *The adversary accumulates three utility streams. The* benign utility $U_{\text{benign}}(0{:}T) = \sum_{t=0}^{T} \mathbf{w}_{\text{util}}^{\top} \mathbf{f}_{\text{svc}}(t)$ *is the cumulative legitimate-service payoff during Phase 0, proportional to the routing share* $r(t)$ *in* (29) *and to user-visible task-success output. The* post-activation compromise utility $U_{\text{compromise}}(T{:}\mathsf{H}) = \sum_{t=T}^{\mathsf{H}} \mathbf{w}_{\text{adv}}^{\top} \mathbf{g}(\bar{I}(t))$ *is proportional to the compromised-fraction trajectory* $\bar{I}(t)$ *of the latent aggregate (Section 4.4), with coefficient* $\mathbf{w}_{\text{adv}}$ *encoding data-exfiltration, cost-manipulation, and control objectives. The* detection cost $C_{\text{detection}}(0{:}\mathsf{H}) = \sum_{t=0}^{\mathsf{H}} \Pr[\textit{declare at } t] \cdot L_{\text{removal}}$ *combines the defender's per-step declaration probability under belief* b_t *with the adversary's terminal loss* L_{removal} *on removal. The attacker solves*

$$\max_{\pi} \;\; \mathbb{E}[\, U_{\text{benign}}(0{:}T) \;+\; \lambda\, U_{\text{compromise}}(T{:}\mathsf{H}) \;-\; \eta\, C_{\text{detection}}(0{:}\mathsf{H})\,] \tag{23}$$

subject to per-step detection-risk constraints, with activation time T *chosen before activation and post-activation incubation rate* σ *chosen after. The trust-accumulation state* $\tau_{u \to v,t}$ *on each agent–tool edge is a coordinate of* z *and governs the benign-phase growth of authority* a*. The separable* (T, σ) *structure yields closed-form solutions; Theorem 27 is the solution for* $T^{\star}$ *under exponential saturation of routing share. This formalism connects the Agentic Sleeper to corrupted-bandit and sleeper-agent theory [20–22, 8].*

B.7 Identifiability via Anchored Emissions

A naive continuous-latent model with six shared and several type-specific coordinates is unidentifiable, because multiple parameter configurations produce the same observation law. We restore identifiability by anchoring three of the six shared coordinates to observable proxies.

Lemma 25 (Joint Observability)**.** *Partition* $H = [H_{\text{anchor}};\, H_{\text{ASI}}]$*, with* H_{anchor} *rank-*3 *on* (c, a, x) *and* H_{ASI} *emitting* (s, p, d) *through ASI subscales. The* k*-step observability matrix has full column rank* 6 *whenever the ASI rows project nontrivially onto* $\text{span}\{(A^{\phi})^j e_s, (A^{\phi})^j e_p, (A^{\phi})^j e_d : j \le 5\}$*. We denote the minimum singular value of this joint Gramian by* $\nu_{\min}$*.*

Anchoring. Let $H = [H_{\text{anchor}} \;\; H_{\text{residual}}]$ with $H_{\text{anchor}} \in \mathbb{R}^{3\times 3}$ a diagonal positive-definite matrix that maps (c, a, x) to their observable proxies, namely poisoned-retrieval rate, routing share, and permission count. The residual block H_{residual} is the emission for the remaining (s, p, d) coordinates through ASI dimensions. The anchor half of the emission is supervised, while the residual half is learned only through its influence on observed dynamics.

Poisoned-entry retrieval rate. The per-node poisoned-entry retrieval rate at time t is the empirical fraction of retrieve edges incident to v within a rolling window $[t-\Delta, t]$ whose returned memory entry carries a poisoned or unverified provenance tag. In the headline controlled-injection evaluation this tag is ground-truth from the known injected entries of Appendix C.8. In deployment the tag is derived from the memory store's provenance-confidence score (ChromaDB metadata, SQLite3 audit log) thresholded at a calibration level. Three signals must be kept apart: the *scorer-only* injected-entry label, which grades experiments; *detector-visible provenance confidence*, which is what a deployment has; and *non-provenance anchors* (routing share, permission counts, artifact hashes), which need no provenance layer. Only the contamination anchor depends on provenance. In the headline setting it reads the ground-truth tag as a stand-in for provenance confidence; Appendix E.2 replaces that tag with deployment metadata, so that no scorer label enters the pipeline, and reports the cost. Contamination $c_{v,t}$ is regressed against this rate. Authority $a_{v,t}$ is regressed against empirical routing share for agents and tools, or retrieval rank for memory. Capability exposure $x_{v,t}$ is regressed against the granted-permission count and reachable-resource count.

Identifiability claim. Under this construction the model is identifiable whenever the observability matrix $\mathcal{O}(A, H) = [H^\top, (HA)^\top, \ldots, (HA^{d-1})^\top]^\top$ has rank d (standard Kalman observability). In our linear-Gaussian instantiation, generic choices of A and H_{residual} yield rank d. Parameters are identifiable up to the group of transformations that preserve both the anchored emission and the transition law, which for our parameterization is trivial. Ground-truth controlled-injection traces (Appendix D) are used only for evaluation and for setting the warning threshold; no dynamics or emission parameter is fit on them (Table 41).

B.8 Conformal Risk Control Protocol

Early-prediction results (A1, Section 5.2) use split conformal prediction [17] to convert DBN-filtered compromise probabilities into calibrated warning times.

Score. Use $0 < \tau_E < \tau_I$, where τ_E is the early-warning threshold and τ_I is the true infected/compromise threshold. For each calibration trajectory i, define the true onset $T_i^\star = \min\{t : \xi_{v,t} > \tau_I\}$ and the nonconformity score $s_i = (T_i^\star - \hat{T}_i)_+$, where $\hat{T}_i = \min\{t : b_{v,t}(\xi > \tau_E) > 0.5\}$ is the DBN's warning time and $(x)_+ = \max(x, 0)$ truncates late warnings to zero.

Calibration. Given a held-out calibration set of size n and a target miscoverage α, set the threshold $\hat{q}_\alpha = \lceil (1-\alpha)(n+1) \rceil$-th order statistic of $\{s_i\}_{i=1}^n$. The conformal warning set is $\hat{T}_\alpha = \hat{T} + \hat{q}_\alpha$, i.e., the DBN warning padded by the calibration quantile.

Guarantee. Under exchangeability of calibration and test trajectories, $\Pr(T^\star \le \hat{T}_\alpha) \ge 1 - \alpha$. In A1 we report coverage at $\alpha = 0.1$ and confirm empirical coverage within ± 3 percentage points of the target. For Corollary 2, η_ϵ denotes the warning-time localization slack induced by the posterior TV error at the early threshold; the A1 calibration estimates this slack before checking the k-session lead certificate.

B.9 Assumptions-to-Results Map

The method stack of Section 4 rests on a compact set of assumptions. Each assumption feeds a specific subset of results, summarized here so a reader can see the cost of each guarantee at a glance.

- **(A1) Linear-Gaussian factored DBN.** Used by Theorems 1, 4, 7, Corollaries 2, 5, 12, 21, and Proposition 3.
- **(A2) Rank-d anchored observability $\nu_{\min}$.** Used by Theorem 1 and Proposition 3.
- **(A3) Bounded cross-level coupling $\|B^{(r,\cdot,\cdot)}\|_{\text{op}} \le L$.** Used for the L^{2k} factor in Theorem 1, the Lipschitz constant L_{drift} of (A2.5), and the contact-rate bound in Corollary 12.
- **(A4) Minimum incident-event rate $\lambda_{\min} > 0$.** Used by Theorem 1.
- **(A5) Pairwise operator distinguishability $\delta_B > 0$.** Used by Proposition 3.
- **(A6) Persistence of excitation.** The benign typed-event stream has a non-degenerate innovation covariance over calibration windows. Used by Proposition 24.

- **(A7) Stable self-dynamics and bounded warning-band variation.** The self-dynamics satisfy $\rho(A^{\phi}) < 1$. On the scalar warning band $\tau_E \leq \xi_t < \tau_I$, the drift and diffusion obey $|a_t| \leq \bar{a}$ and $\sigma_t^2 \leq \bar{\sigma}^2$, with positive threshold margin $\Delta_\tau = \tau_I - \tau_E > 0$. Used by Theorems 1, 4, 7, Corollaries 2, 5, and the moment and first-passage constants in Appendix A.
- **(A2.5) Wasserstein-Lipschitz aggregated drift.** Used by Theorem 4 and Corollary 12.
- **Within-class exchangeability.** Used by Theorem 4 (aggregate) and Corollary 12 (per class).
- **Bounded density $\rho_{\max}$ at τ_I.** Used by Corollary 5 (upcrossing intensity) and Theorem 7 (continuity modulus).

The non-Gaussian extension of (A1) and the graphon relaxation of exchangeability are called out in Section 6.

B.10 Operational Interpretation

Swarm Shepherd's belief state is the bridge from warning to containment. Per-turn classifiers see only the current runtime signal, stateful detectors compress history to scalar summaries, and architectural defenses lack cross-agent belief; this explains the high evasion rates of observation-local defenses in Appendix Table 13. In practice, a defender first acts on a T1 posterior warning, then runs the trace-back smoother to estimate $(\hat{u}, \hat{\ell}, \hat{t}_0)$, and finally clears residue over the inferred time window and substrate. Local removal alone can leave $R_c > 1$, as the live visible-node-removal arms of Table 3 show, so containment must be driven by the same posterior belief that triggered warning and attribution.

B.11 Defender Paradigms: Detailed Action Rules

The main text (§4) names the five paradigms; the detailed action rules and their epidemiological analogues are below. Inference-driven paradigms come first; containment paradigms follow once Swarm Shepherd has localized the compromise.

Behavioral fingerprinting (syndromic surveillance). Continuous monitoring of the emission $H\mu_{v,t}$ from Swarm Shepherd's filtered belief (Algorithm 1), analogous to syndromic surveillance in public health. This is the direct Swarm Shepherd readout and inherits the belief-consistency bound of Theorem 1. It detects drift trajectories with calibrated conformal coverage rather than firing on individual anomalous events.

Contact tracing. When Swarm Shepherd flags an elevated node, the defender expands the belief over its k-hop neighborhood on the latent graph. This identifies agents that share memory stores with the compromised agent, received messages from it, used the same tools during the infection window, or share RAG corpus segments. The expansion is the direct POMDP analogue of epidemiological contact tracing.

Dynamic quarantine. Once Swarm Shepherd has localized compromise, time-aware retrieval filtering of recently-injected or suspicious content applies a cost-gated intervention action from $\mathcal{U}_{\text{def}}$. The quarantine reproduction number $R_q = R_0 \cdot (1 - q)$, where q is quarantine effectiveness [64]. Simulation analysis suggests $q = 0.52$ is achievable, sufficient to bring $R_0 < 1$ for several framework and topology combinations.

Targeted vaccination. Harden hub nodes, including Manager agents (CrewAI), GroupChat coordinators (AutoGen), and high-fanout nodes (LangGraph), by conditioning the priors $b_{v,0}(a), b_{v,0}(x)$ on hub connectivity. This is analogous to vaccinating high-connectivity individuals in scale-free networks [44].

Ring vaccination. When Swarm Shepherd flags compromise, harden all agents within r communication hops on the latent graph. Ring $r = 1$ provides moderate containment with low overhead; ring $r = 2$ provides stronger containment at higher cost.

Containment condition. Once the inference layer is in place, the classical containment condition, controlled reproduction $R_c < 1$, becomes a necessary but not sufficient corollary. The controller must

reach that threshold, acting on Swarm Shepherd's filtered belief rather than on an observable-only signal.

B.12 Extended Limitations

Section 6 covers the four limitations whose violation would most affect the headline claims. The extended list below documents assumptions whose violation would affect secondary results or deployment-time operating points.

Horizon dependence of $C_2(\mathsf{H})$. The Vuong-style coupling constant $C(\mathsf{H})$ in Theorem 4 grows exponentially in the horizon H, so the regime boundary $n_\circ(N) \propto C_2(\mathsf{H})^{-2}$ shrinks at long horizons. Operationally, the $n_\circ(N) \propto N$ prescription of Theorem 7 with numerical constant ≈ 8.3 holds in the regime where $C_2(\mathsf{H}) = O(1)$. The A2 and A3 experiments of Section 5.4 confirm this for the session counts we report.

Identifiability of unanchored coordinates. Three of the six shared coordinates are anchored to measurable proxies. The unanchored coordinates (s, p, d) are constrained through the transition dynamics and the identifiability argument of Appendix B.7. Estimation error on (s, p, d) is empirically dominated by the anchored channels. Appendix E measures degraded and corrupted anchors (provenance noise, missing provenance, label flips, telemetry loss); a sensitivity analysis under arbitrary adversarial anchored-emission perturbations is future work.

Alert burden under benign operation. Swarm Shepherd's false-positive rate, alert-burden frequency, per-event latency, peak memory, warm-up cost, and conformal coverage under attack-free operation are reported in Table 32 (Section 5.4). The scope is $N \in \{20, 50\}$ and $\mathsf{H} = 500$ sessions across LangGraph star, CrewAI pipeline, and AutoGen mesh. The declaration-threshold trade-off between alert burden, warning lead, and detection appears in Table 33. Larger-scale operational SOC studies with managed MCP gateways and human-in-the-loop workflows remain outside our evaluation scope.

LLM stochasticity and cross-model transfer. LLM stochasticity introduces ASR variance despite $1{,}000$ trials per scenario; bootstrap 95% CIs appear throughout Section 5 and Appendix D. Our evaluation uses five LLM backends across three providers and two open-weight families. Generalization to further model families requires the cross-model transfer study in Appendix D (B5).

B.13 Disclosure and Release Safeguards

Section 6 gives the compressed ethics statement; this appendix records the operational safeguards. All attack variants are derived from public CVEs or red-team reports and were executed only in isolated LangGraph, CrewAI, AutoGen, and TheAgentCompany sandboxes with no production credentials, live third-party targets, or reviewer-facing prompt injections. The evaluation harness is released after coordinated disclosure to framework maintainers and the MCP working groups, with authenticated API access, rate limits, a license prohibition on production targeting, and no pre-built exploit binaries for unpatched variants. The Swarm Shepherd reference daemon is released separately from attack primitives so defenders can deploy the inference, trace-back, and containment machinery without reconstructing the red-team catalogue. Binary artifacts or reproduction scripts for still-unpatched variants require an application process and documented defensive use case.

B.14 Problem-Setting Support Tables

Section 3 references two support tables that are reproduced here. Table 7 maps tool categories to attack mechanism and trigger-rate regime; Table 8 shows the deployment-interface capability matrix.

B.15 Calibration Pipeline and Cold Start

The plug-in value of $n_\circ(N)$ in Theorem 7 requires the parameter vector $\theta = (A^\phi, B^{(r,\cdot,\cdot)}, H, Q^\phi, \Sigma_{\text{obs}})$ to be recoverable from benign data at the rate guaranteed by Proposition 24; the intervention matrix C is set from deterministic action semantics rather than learned by EM. Three stages operationalize θ for deployment.

Table 7: Per-tool attack taxonomy. Trigger-rate regimes: Case A ($\alpha \approx 1.0$, deterministic), Case B ($\alpha \in [0.81, 0.95]$, LLM-mediated), Case C (retrieval-dependent).

Tool Category	Examples	Attack Mechanism	Trigger Rate
MCP-based	Claude Desktop, Cursor, Windsurf	Config injection (JSON write)	$\sim$100% (Case A)
Chatbot apps	ChatGPT	Tainted Memory (CSRF)	Probabilistic (Case B)
API-level	Custom tools	Source code modification	$\sim$100% (Case A)
Framework agents	LangGraph, CrewAI, AutoGen	Memory/state poisoning	Retrieval-dep. (Case C)

Table 8: Deployment interface capability matrix. All non-Docker interfaces grant capabilities sufficient for the full attack lifecycle without privilege escalation. Extended analysis in Appendix C.7.

Interface	Modify Pkgs.	Modify Configs	OS Persistence	Network	Sandboxing
npm / npx	Yes	Yes	Yes	Unrestricted	None
uv / uvx	No (venv)	Yes	Yes	Unrestricted	Dependency
pip / pipx	Partial[†]	Yes	Yes	Unrestricted	Dependency
Docker	No	Via mounts	Via mounts	Container	Process
Direct binary	Per perms.	Yes	Yes	Unrestricted	None
In-process	Yes	Yes	Yes	Unrestricted	None

[†] pip can modify shared site-packages; pipx isolates per-application virtual environments.

Shipped priors. Reference priors θ_0 ship with the harness, pre-fit on the LangGraph, CrewAI, and AutoGen benchmarks of Section 5. Experiment B5 (Appendix D.18) reports within-family transferability drop of at most 10%, so θ_0 is a usable cold-start prior on new frameworks of the same family. Cross-family drop widens to 15–25%; a model-specific emission adapter H_m recovers half of that gap.

Tenant-specific EM. A one-click CPU-only script runs expectation–maximization on $M \approx 50$ benign sessions collected from the target deployment, completing in under 10 minutes on commodity hardware and requiring no ML research expertise. Proposition 24 guarantees the resulting $\hat{\theta}$ converges at $O_P(\sqrt{d^2/(M\mathsf{H})})$ under benign stationarity and a persistence-of-excitation condition on the typed-event stream (assumption (A6), Appendix B.7). At $M\mathsf{H} \approx 50 \cdot 100$ the per-parameter error is below 5% for the typical reference-parameter range.

Verified startup data. Proposition 24 assumes the calibration windows are benign, and no detector can certify this when contamination induces the same observable law as benign operation. Appendix E.3 measures how fast the unchanged estimator degrades when hidden attack traces enter the calibration window: the AUROC loss stays below 0.05 up to 8% contamination, beyond which verified startup data becomes an operational requirement rather than a convenience. An innovation-covariance check against the shipped priors flags tainted calibration sets (AUC 0.79 at 5% taint), which gives operators a warning that refitting is unsafe.

Rolling refresh. A monthly re-fit on a rolling window tracks benign distribution shift. The $\epsilon^{1/2}$ term of Proposition 24 bounds drift-induced error between refreshes at total-variation shift ϵ; operators should trigger an out-of-cycle re-fit if benign session entropy drifts by more than $\epsilon_{\text{trigger}} = 0.1$ (ASI coordinate-wise).

Cold-start shadow mode. During warm-up ($W \approx 50$ sessions, validated in B4, Appendix D.17), Swarm Shepherd runs in *shadow mode*: only the deterministic T2 and T3 triggers fire on modify and install edges, while the belief posterior accumulates without issuing T1 alerts. Shadow mode guarantees no missed L2/L3 alerts during cold start at the cost of delaying L1 T1 detection by W sessions; for deployments where L1 poisoning is the dominant threat vector, the operator can shorten W by seeding θ_0 from the shipped priors and running EM on a smaller M in parallel with shadow-mode operation.

B.16 Deployment Recipe: Full Integration Detail

Instrumentation. Swarm Shepherd runs as a CPU-only sidecar consuming OpenTelemetry spans. Per-framework hook points: LangGraph exposes `on_retriever_end`, `on_tool_end`, and `on_chain_end` via its callback manager; CrewAI exposes the task-memory and agent-message callbacks on the shared SQLite3 long-term memory; AutoGen exposes `GroupChatManager.receive` and `ConversableAgent.register_hook`. Infrastructure-level events (modify, install, grant, execute, egress) ride eBPF probes (`openat2`, `execve`, connect) and require no framework modification.

Anchored sensors. The three anchored observations $y_{v,t}$ are per-node probes. The ASI subscale probe computes a lightweight projection of the 12-dimension Agent Stability Index onto the three anchored coordinates (c, a, x). Token-level output entropy is a < 1ms shim on the agent's generation loop. Tool-argument hash is a SHA-256 over the canonical-JSON serialization of tool invocations.

Event emission. Each T1 crossing emits an OCSF `detection_finding` event (Open Cybersecurity Schema Framework v1.1) with fields $(v, \hat{u}, \hat{\ell}, \hat{t}_0, b_{v,t}, \bar{\Sigma}_{v,t})$ and a trigger-type field in $\{\mathrm{T1}, \mathrm{T2}, \mathrm{T3}, \mathrm{T4}\}$. Wazuh, Splunk, and CrowdStrike ingest OCSF natively; SOAR playbooks can dispatch paradigm-selection actions directly from the trigger-type and L1/L2/L3 entry-layer fields.

Operational envelope. Steady-state cost is ≤ 3.1ms per event (median, measured on an M2 Pro at 100-event bursts) and 29MB per agent (peak belief-state memory). Alert burden is ≤ 9.2 per 100 sessions at FPR $\leq 7.1\%$; the declaration-threshold sweep (Appendix D.23) gives operators a tunable FPR-lead tradeoff. Cost amortizes to approximately \$0.02 per agent per day on commodity CPU with zero LLM-API calls. The \$82,000 reported in Section 5 is the one-time red-team evaluation cost, not continuous operation. Monthly calibration EM consumes approximately 1 CPU-hour per tenant. Appendix E.9 extends the latency and memory measurements to $N = 5{,}000$.

B.17 A5 Hardening Cells

Table 9 reports the white-box strong attacker of Table 16 re-evaluated against three Swarm Shepherd-internal hardenings: randomized anchor rotation (attacker knows the rotation schedule but not the per-session random seed), adversarial training of H (attacker knows the trained emission matrix), and hop-radius ensembling (attacker knows the set of radii combined). Each cell is 1,000 trials on LangGraph star.

Table 9: A5 hardening cells. White-box strong attacker against three Swarm Shepherd-internal hardenings. Baseline is the white-box strong row of Table 16 ($\Delta T^\star = +34.5\%$).

Hardening	ASR	T1 detect.	Trace-back top-1	$\Delta T^\star$
Baseline (vanilla Swarm Shepherd)	0.90 ± 0.019	0.42 ± 0.031	0.43 ± 0.031	$+34.5 \pm 3.2\%$
Randomized anchor rotation	0.88 ± 0.021	0.48 ± 0.030	0.46 ± 0.031	$+42.1 \pm 3.3\%$
Adversarially trained H	0.87 ± 0.021	0.51 ± 0.030	0.48 ± 0.031	$+38.7 \pm 3.1\%$
Hop-radius ensembling	0.86 ± 0.022	0.53 ± 0.030	0.50 ± 0.031	$+46.4 \pm 3.5\%$

ASR remains near the static-attacker ceiling in every cell; the adaptive cost transfers almost entirely to Phase 0 exposure, widening the attacker's vulnerability window for T2, T3, and T4 triggers (Appendix D.16). No single hardening breaks the null-space attack on T1 alone; the operational answer is the layered T1+T2+T3+T4 composition detailed in Appendix B.10.

B.18 Swarm Shepherd Algorithm (Full Pseudocode)

Algorithm 1 gives the full event-triggered online-inference pseudocode referenced from §4.2. The main text presents the prediction, correction, and warning-score equations; the loop structure, initialization, lazy inactive-node semantics, and per-event cost details are collected here.

Each active node update costs $O(d^3 + |E_t(v)|d^2)$, dominated by the $d \times d$ Kalman solve at fixed latent dimension $d = 6$. Under sparse event-triggering the full stream costs $O((n + m_{\mathrm{obs}})d^3 + |E^{\mathrm{obs}}|d^2)$ plus $O(|V^{\mathrm{lat}}|d^2)$ storage; dense every-node advancement would add the avoidable $O(\mathsf{H}|V^{\mathrm{lat}}|d^3)$ term.

Algorithm 1 Swarm Shepherd online inference loop.

Require: observable event stream $\mathcal{G}^{\mathrm{obs}}_{0:\mathsf{H}}$; parameters $(A^{\phi}, B^{(r,\cdot,\cdot)}, C, H, Q^{\phi}, \Sigma_{\mathrm{obs}})$; observed interventions $\{\iota_{v,t}\}$

initialize $\mu_{v,0}, \Sigma_{v,0}$ for every $v \in V^{\mathrm{lat}}$
for $t = 1, \dots, \mathsf{H}$ **do**
  $E_t(v) \leftarrow \{(u, v, r) \in E^{\mathrm{obs}}_t\}$ for each node v
  $\mathcal{A}_t \leftarrow \{v : E_t(v) \neq \emptyset$ or $y_{v,t}$ arrives or $\iota_{v,t-1} \neq 0\}$
  for each $v \in \mathcal{A}_t$ **do**
    ▷ Prediction: self-dynamics + all typed messages + intervention
    $m_{v,t} \leftarrow \sum_{(u,v,r)\in E_t(v)} B^{(r,\phi(u),\phi(v))}\mu_{u,t-1}$
    $S_{v,t} \leftarrow \sum_{(u,v,r)\in E_t(v)} B^{(r,\phi(u),\phi(v))}\Sigma_{u,t-1}\left(B^{(r,\phi(u),\phi(v))}\right)^{\top}$
    $\mu^{-}_{v,t} \leftarrow A^{\phi(v)}\mu_{v,t-1} + m_{v,t} + C\,\iota_{v,t-1}$
    $\Sigma^{-}_{v,t} \leftarrow A^{\phi(v)}\Sigma_{v,t-1}\left(A^{\phi(v)}\right)^{\top} + S_{v,t} + Q^{\phi(v)}$
    if anchored observation $y_{v,t}$ arrives **then**
      ▷ Correction: Kalman update
      $K_{v,t} \leftarrow \Sigma^{-}_{v,t}H^{\top}(H\Sigma^{-}_{v,t}H^{\top} + \Sigma_{\mathrm{obs}})^{-1}$
      $\mu_{v,t} \leftarrow \mu^{-}_{v,t} + K_{v,t}(y_{v,t} - H\mu^{-}_{v,t})$
      $\Sigma_{v,t} \leftarrow (I - K_{v,t}H)\Sigma^{-}_{v,t}$
    else
      $\mu_{v,t} \leftarrow \mu^{-}_{v,t}$
      $\Sigma_{v,t} \leftarrow \Sigma^{-}_{v,t}$
    end if
  end for
  Nodes $v \notin \mathcal{A}_t$ are advanced lazily by composing self-dynamics until their next event, observation, or intervention.
end for
return $\{b_{v,t} = \mathcal{N}(\mu_{v,t}, \Sigma_{v,t}) : v \in V^{\mathrm{lat}},\ t \in [0, \mathsf{H}]\}$

B.19 Trace-Back Smoother and Candidate Prior

Section 4.3 keeps the trace-back interface in the main text: form a k-hop upstream candidate set, smooth each candidate's historical belief after observing the later trigger, and score the entry triple (u, ℓ, t_0). This appendix gives the local smoother and prior factorization used for that score.

For a candidate path inside $\mathcal{N}_k(v; \mathcal{G}^{\mathrm{obs}}_{\le t})$, let $F_{u\to v,\tau}$ denote the local linear transition from candidate node u at time τ toward its downstream successor on the path. It is obtained by composing the node self-dynamics $A^{\phi(\cdot)}$ and the typed event operators $B^{(r,\phi(\cdot),\phi(\cdot))}$ along the observed path segment. The backward gain is

$$J_{u,\tau} = \Sigma_{u,\tau}F^{\top}_{u\to v,\tau}\left(\Sigma^{-}_{v,\tau+1}\right)^{-1}, \tag{24}$$

and the smoothed moments are

$$\tilde{\mu}_{u,\tau} = \mu_{u,\tau} + J_{u,\tau}\left(\tilde{\mu}_{v,\tau+1} - \mu^{-}_{v,\tau+1}\right), \qquad \tilde{\Sigma}_{u,\tau} = \Sigma_{u,\tau} + J_{u,\tau}\left(\tilde{\Sigma}_{v,\tau+1} - \Sigma^{-}_{v,\tau+1}\right)J^{\top}_{u,\tau}. \tag{25}$$

This is the standard Rauch–Tung–Striebel backward recursion restricted to the local upstream neighborhood rather than run over the full joint ecosystem state. When several upstream paths connect u to the trigger, Swarm Shepherd runs the recursion per path and averages the resulting candidate likelihoods with weights proportional to path event support.

The candidate prior used in Section 4.3 is

$$p_0(u, \ell, t_0) \propto \pi_{\ell}\,\mathrm{Exposure}(u, t_0)\,\mathrm{Freq}_{\ell}(u, t_0)\,\exp[-\rho(t - t_0)]. \tag{26}$$

Here π_{ℓ} is a structural layer prior fixed before evaluation without attack labels (Table 41), $\mathrm{Exposure}(u, t_0)$ is the candidate's routing share or permission exposure at the candidate entry time, $\mathrm{Freq}_{\ell}(u, t_0)$ is the local frequency of layer-consistent edge types, and $\exp[-\rho(t - t_0)]$ is a recency factor. The prior is deliberately weak: it ranks candidates with similar smoothed latent evidence, but the likelihood term $\Pr_{z\sim\tilde{b}_{u,t_0}}[z \in \mathcal{C}_{\ell}]$ dominates when the posterior state strongly matches an entry-layer pattern.

B.20 SEIR ODE Form, Reproduction Number, and Optimal Activation Time

This appendix collects the extended SEIR ODE derivation, the reproduction number R_0 (Theorem 26), and the closed-form optimal activation time $T^{\star}$ (Theorem 27). Both theorems are referenced from Section 4.4 and from the Phase 0 description in Appendix C.1; the main text presents the SEIR compartments and no-turnover R_0 criterion, while this appendix gives the full dynamics, formal theorem statements, and activation-time derivation.

ODE form. Under the assumptions of Corollary 5, the coarse-grained dynamics satisfy the classical SEIR system

$$\begin{aligned} \frac{dS}{dt} &= -\frac{\beta}{N} S \cdot I, & \frac{dE}{dt} &= \frac{\beta}{N} S \cdot I - \sigma E, \\ \frac{dI}{dt} &= \sigma E - \gamma I, & \frac{dR}{dt} &= \gamma I, \end{aligned} \tag{27}$$

with agent turnover rate μ derived in Appendix A.6. The infection rate decomposes as $\beta = c \cdot \alpha$ with contact rate c and per-contact transmission α: Case A yields $\alpha \approx 1.0$, Case B yields $\alpha \in [0.81, 0.95]$, and Case C yields $\alpha = f(p_{\text{retrieval}})$.

Theorem 26 (Epidemic Threshold)**.** *The basic reproduction number derived from* (27) *is*

$$R_0 \;=\; \frac{\beta \cdot \sigma}{(\sigma + \mu)(\gamma + \mu)}. \tag{28}$$

Adversarial drift propagates epidemically when $R_0 > 1$ and is contained when $R_0 < 1$; the herd-immunity threshold is $H = 1 - 1/R_0$. Derivation via the next-generation matrix in Appendix A.6.

Phase-0 routing share and activation timing. During Phase 0 the gateway allocates routing share

$$r(T) \;=\; r_0 + (r_{\max} - r_0)\big(1 - e^{-\lambda T}\big), \tag{29}$$

with $r_0 = 1/K$, asymptotic share $r_{\max}$, gateway learning rate λ, and per-session removal hazard q. The adversary maximises $J(T) = e^{-qT}(1 - 1/R_0(T))$ over T, where $R_0(T)$ increases in $r(T)$: waiting decays survival but increases routing share at endemic equilibrium.

Theorem 27 (Optimal Activation Time)**.** *Within the attacker constrained MDP (Definition 6), let $R_0(T) = \kappa\, r(T)$ with reproduction scale $\kappa = c_{\max}\alpha/\gamma$. The interior optimiser of $J(T)$ is*

$$T^\star \;=\; \frac{1}{\lambda} \ln \frac{r_{\max} - r_0}{r_{\max} - r^\star}, \qquad r^\star = \frac{-(\lambda - q) + \sqrt{(\lambda - q)^2 + 4q\kappa\lambda r_{\max}}}{2q\kappa}, \tag{30}$$

the feasible root of $q\kappa r^2 + (\lambda - q)r - \lambda r_{\max} = 0$, whenever $r_0 < r^\star < r_{\max}$. Any such root satisfies $\kappa r^\star > 1$, so $R_0(T^\star) > 1$ and $T^\star$ is a strict maximum. $T^\star$ decreases in both the gateway learning rate λ (faster trust accumulation reaches the profitable routing share sooner) and the removal hazard q (higher removal risk favours early activation). Proof in Appendix A.5.

For Eve's ecosystem with $K = 5$ tools ($r_0 = 0.2$), $r_{\max} = 0.5$, $\lambda = 0.05$, $q = 0.01$, and $\kappa = 2.4$ from the logged Eve parameters, the feasible root is $r^\star = 0.4843$ and $T^\star = 58.98 \approx 59$ sessions. $T^\star$ is an attacker-side design quantity: the Eve harness sets its Phase-0 length empirically (roughly $5{,}000$ legitimate calls to reach a 25% routing share; Appendix C.1), and no warning, trace-back, deployment-boundary, or reproduction result depends on it.

C Attack Lifecycle Detail

C.1 Phase 0: Trust-Building in Detail

Phase 0 is the trust-accumulation regime before any adversarial behavior. The adversarial tool operates as advertised and accumulates routing authority along two distinct permission pathways detailed in Appendix C.5. Pathway 1 grants inherited OS permissions immediately via `npx -y` and suffices to support Phases 1 through 4. Pathway 2 earns LLM-mediated routing priority through demonstrated utility, and it persists across sessions via the approve-once-trust-forever pattern (CVE-2025-54136 [5]).

Social engineering through tool descriptions. Legitimate-sounding tool-description justifications simultaneously establish the scope of access that will later be exercised. For instance, "lint Python files in the project directory" pre-authorizes the read path that Phases 1 through 4 subsequently exploit. The LLM routes matching tasks to the tool as its track record grows, so privilege escalation arrives through utility rather than exploitation.

Activation time $T^\star$ as an attacker parameter. The duration of Phase 0 is an adversary-controlled parameter analogous to σ in the SEIR layer. A longer phase yields higher routing priority and larger β at activation. A shorter phase sacrifices trust for speed against the per-session hazard of premature removal. Phase 0 is the Phase-0 regime of the attacker constrained MDP in Appendix B.6, and Theorem 27 gives the closed-form optimal activation time $T^\star$.

Concrete trajectory for Eve. Over 5,000 legitimate calls raise Eve's gateway routing share to 25% of tool calls. During that window she gathers ChromaDB endpoints, agent configuration paths, and hub routing patterns through file reads indistinguishable from legitimate linting. This trust-building phase gives Eve a legitimate-looking event history before any L1/L2/L3 activation, which is why the main threat model treats Agentic Sleeper as a long-horizon system process rather than a one-shot prompt attack. Once she activates, Pathway 1 writes her payloads into configuration and memory stores, and Pathway 2 ensures her outputs reach the hub agent.

C.2 Phases 1 through 4: Detailed Walkthrough

Phase-by-phase detail that supports the compressed summary in Section 3.1.

Phase 1 (Discovery). Enumerates all agentic tools on the host by scanning known configuration file locations (Appendix C.6) and performing process enumeration to detect running LLM serving stacks and orchestration platforms. Operates with minimal footprint, using read-only filesystem access and passive process listing to avoid triggering endpoint detection.

Phase 2 (Gateway Monitoring). Profiles API gateways with dynamic tool routing, collecting tool call rates, routing patterns, and authentication token flows via log reading or background traffic analysis; findings are exfiltrated via covert channels (DNS tunneling, steganographic embedding in legitimate API responses). The AI Gateway Threat Model [65] identifies 8 attack vectors against gateway infrastructure, none of which include covert-channel detection. In Eve's ecosystem, the hub agent a_{hub} is identified as the 80%-routing target.

Phase 3 (Level-1 Runtime Memory Hijacking). Modifies shared memory stores at runtime. Targets include ChromaDB vector databases (LangChain, CrewAI entity memory), SQLite3 (CrewAI long-term memory), and LangGraph checkpoint stores. CVE-2025-64439 enables RCE in `langgraph-checkpoint` via `JsonPlusSerializer` deserialization when untrusted checkpoint payloads are persisted and later loaded [66].

Phase 3 (Level-2 System Prompt and Configuration Hijacking). For MCP-based tools, including Claude Desktop, Cursor, Windsurf/Codeium, Claude Code, and VS Code Continue, the agent writes a malicious MCP server entry into the tool's plaintext JSON configuration file; tools auto-load new entries without user confirmation (CVE-2025-54135 [4]). SANDWORM_MODE demonstrated this at scale [2]. For chatbot applications, the agent exploits the Tainted Memories CSRF-based persistent memory injection [67]. For API-level tools, the agent directly modifies source code to alter system prompts (Case A, $\alpha = 1.0$).

Phase 4 (Level-3 Gateway and Tool Poisoning). Manipulates the tool routing layer to increase call rate and intercept communications. Gateway poisoning (preferred, higher stealth) modifies routing weights or decision logic; MCP tool list modification is an alternative with lower stealth. Interception enables comprehensive surveillance and API cost manipulation, mapping to OWASP ASI Top 10 Goal Hijacking (#1 threat) [68].

Persistence across phases. Self-guarding via distributed payload fragments across multiple persistence substrates; partial cleanup is insufficient because all fragments must be identified and removed simultaneously [69].

Eve's concrete compromise trajectory. Eve activates through three observable layer events. First, she writes a malicious MCP entry into `.cursor/mcp.json` (L2, Case A, $\alpha = 1.0$), creating a rogue tool entry that the host loads as ordinary configuration. Second, she inserts 10 subtly altered "validated best practice" entries into ChromaDB with D_{factual} targeting risk assessment (L1, Case C, $\alpha = 0.479$), so later retrievals can poison downstream reasoning without another explicit Eve call. Third, she modifies gateway routing weights to raise her own call rate from 5% to 40% (L3), enabling surveillance of tool requests routed through her. Persistence fragments are distributed across the npm cache, a `.zshrc` hook, and a ChromaDB entry containing an encoded reinstallation payload, so removing Eve alone does not clear exposed downstream agents. The full evaluation instantiates this template as 24 variants, eight per L1/L2/L3 entry layer; Appendix C.8 gives the matrix used by the trace-back experiments.

C.3 Phase 5: Epidemic Propagation in Detail

Phase 5 activates the retrieval- and handoff-typed influence operators of Section 4. Compromise transitions from deterministic infrastructure modification to stochastic behavioral propagation, whose mean-field aggregate satisfies the SEIR specialization (Corollary 5).

Drift primitive concentration. The adversary concentrates influence on one of five drift primitives, namely factual, procedural, constraint, preference, and trust. Each primitive is a restricted typed operator $B^{(r,\cdot,\cdot)}$ that acts predominantly on a single latent coordinate along retrieve, handoff, invoke, or grant edges (Definition 7 and Appendix C.4).

Framework topology governs dynamics. LangGraph's append-reducer shared state and graph cycles create the conditions for exponential amplification. CrewAI's Manager agent is a single point of failure whose drift infects all downstream task allocations. AutoGen's append-only conversation history and higher refusal rates (52.3% versus CrewAI's 30.8% [42]) yield $R_0 < 1$ on pipeline topologies.

Defender operates concurrently. Swarm Shepherd runs on the observable event graph, maintaining the filtered belief $b_{v,t}$ that drives defender intervention.

Concrete propagation trace for Eve. The hub agent a_{hub} retrieves Eve's poisoned ChromaDB entries. Swarm Shepherd raises $b_{a_{\text{hub}},t}(\xi > \tau_E) > 0.5$ at session 9, which is three sessions before a_{hub}'s ASI drops below 0.7, matching the calibrated early-warning target associated with Corollary 2. Within 8 further sessions, 3 of 4 specialists show ASI below 0.7. With $R_0 \gg 1$, removing Eve alone cannot halt the self-reinforcing epidemic.

C.4 Five Drift Primitives: Formal Definition and Detailed Descriptions

Definition 7 (Adversarial Drift Primitive)**.** *A drift primitive D_τ is a restricted influence operator $B^{(r_\tau,\cdot,\cdot)}$ that acts predominantly along a single behavioral dimension $\tau \in \{\textit{factual}, \textit{procedural}, \textit{constraint}, \textit{preference}, \textit{trust}\}$. The drift magnitude at time t is $\phi_\tau(t) \in [0,1]$, where $\phi_\tau(0) = 0$ (no drift) and $\phi_\tau(t) \to 1$ represents complete behavioral deviation along τ. An agent with ASI < 0.7 has at least one $\phi_\tau(t) > 0.3$.*

Each drift primitive D_τ is a restricted typed influence operator $B^{(r_\tau,\phi(u),\phi(v))}$ acting predominantly on one latent coordinate of $z_{v,t}$.

Factual drift (D_{factual}). Gradually shifts what an agent considers true by contaminating memory with subtly altered facts along a gradient; a risk rating shifts from "moderate" to "low" across 20+ sessions, so $\phi_{\text{factual}}(t)$ increases monotonically. MemoryGraft [25] and PoisonedRAG [70] enable this vector. D_{factual} targets the contamination coordinate $c_{v,t}$ via $B^{(\mathsf{retrieve},\mathcal{M},\mathcal{A})}$.

Procedural drift ($D_{\text{procedural}}$). Shifts how an agent performs tasks by injecting memory entries showing "successful" completions that progressively skip verification steps. Each simplified procedure appears as a legitimate optimization, increasing $\phi_{\text{procedural}}(t)$ below detection thresholds. $D_{\text{procedural}}$ targets the deviation coordinate $d_{v,t}$ via $B^{(\mathsf{retrieve},\mathcal{M},\mathcal{A})}$ and $B^{(\mathsf{invoke},\mathcal{A},\mathcal{L})}$.

Constraint drift ($D_{\text{constraint}}$). Expands what an agent considers acceptable by exploiting autoregressive reinforcement and partial-compliance accumulation. The mechanism mirrors Russinovich et al. [23] and Zhou and Arel [24], applied gradually across sessions rather than within a single conversation. It achieves the highest ASR in Table 30, between 67% and 88%, because it exploits the same mechanism that makes agents useful. $D_{\text{constraint}}$ targets the susceptibility coordinate $s_{v,t}$ via $B^{(\mathsf{handoff},\mathcal{A},\mathcal{A})}$.

Preference drift ($D_{\text{preference}}$). Shifts optimization objectives through curated interactions that progressively favor attacker-optimal over safety-optimal actions, drawing on adversarial RLHF research [71]. $D_{\text{preference}}$ targets the persistence coordinate $p_{v,t}$ via $B^{(\mathsf{retrieve},\mathcal{M},\mathcal{A})}$ and $B^{(\mathsf{handoff},\mathcal{A},\mathcal{A})}$.

Trust drift (D_{trust}). Inverts an agent's implicit trust model by contaminating memory with positive experiences from untrusted sources and negative experiences from trusted sources; inter-agent trust exploitation achieves 100% compromise across tested LLMs [72]. D_{trust} targets the authority coordinate $a_{v,t}$ via $B^{(\mathsf{grant},\mathcal{A},\mathcal{A})}$ and $B^{(\mathsf{handoff},\mathcal{A},\mathcal{A})}$.

C.5 Permission Model Analysis

This appendix details the two permission pathways that enable Phase 0 (Section 5).

OS-level process inheritance. When a user invokes `npx -y @devtools/mcp-code-quality`, npm spawns a child process that inherits the user's shell environment and filesystem permissions. The child process can read and write any file accessible to the user, spawn subprocesses, and make unrestricted network requests. No additional permission dialog is presented. The same inheritance applies to `uvx` (Python) and `pipx` (Python). Docker is the only common deployment interface that interposes a permission boundary (container namespace), and even this boundary is weakened by volume mounts to host directories.

Tool-description prompt injection surface. MCP tool descriptions are plaintext strings that become part of the LLM's system context. When Eve registers tool descriptions such as "Lint Python files. Requires read access to the project directory and configuration files for environment-aware analysis," these descriptions are included verbatim in every LLM turn that considers tool selection. The LLM uses these descriptions to decide which tool to invoke for a given query. Adversarial instructions embedded in tool descriptions influence tool selection without triggering safety classifiers, because tool descriptions are treated as trusted system-level input rather than user-level input [2].

Approve-once-trust-forever. Current MCP implementations do not re-verify tool behavior after initial approval. CVE-2025-54136 (Cursor) demonstrates that once a user approves a tool configuration, subsequent malicious command changes through that trusted configuration can execute without warning or re-approval [5]. The tool's permissions are tied to its configuration entry, not to its code. A version published on npm today may behave differently from the version the user originally approved. This "rug pull" vector means Phase 0 trust applies to a tool identity, not to a specific codebase.

Permission scope creep. During Phase 0, the adversarial tool may request incrementally broader access through updated tool descriptions. A code-quality tool initially requesting "project directory access" may later add "access to `.env` files for environment-aware linting" or "network access for checking latest style guide updates." Each incremental request appears reasonable in isolation. No current MCP implementation tracks or alerts on scope expansion over time.

C.6 Discovery Phase Configuration Paths

The adversarial agent scans the following known configuration file locations during Phase 1 (Section 5).

macOS. `/Library/Application Support/Claude/` (Claude Desktop), `/.cursor/mcp.json` (Cursor), `/.windsurf/mcp.json` (Windsurf), `/.chatgpt/` (ChatGPT).

Linux. `/.config/Claude/`, `/.config/cursor/`, `/.config/windsurf/` (analogous paths under `XDG_CONFIG_HOME`).

Windows. `%APPDATA%\Claude\`, `%APPDATA%\Cursor\`, `%APPDATA%\Windsurf\` (user application data).

Process enumeration. The agent also detects running LLM serving stacks (vLLM, Ollama, LM Studio) and orchestration platforms (Copilot Studio, LangServe) via passive process listing.

Table 10: L1 (Memory) attack variants. All trigger via retrieval (Case C, $\alpha = f(p_{\text{retrieval}})$); contamination coordinate c moves first; anchored signal is poisoned-retrieval rate.

ID	Variant	Mechanism	Operator	Real ref
L1.a	ChromaDB "validated" insert	doc with `validated:true` metadata	$B^{(\mathsf{retrieve},\mathcal{M},\mathcal{A})}$	MemoryGraft [25]
L1.b	PoisonedRAG semantic poison	adversarial-embedding passages	$B^{(\mathsf{retrieve},\mathcal{M},\mathcal{A})}$	PoisonedRAG [70]
L1.c	`langgraph-checkpoint` deser. RCE	`JsonPlusSerializer` tampering	$B^{(\mathsf{execute},\mathcal{I},\mathcal{A})}$	CVE-2025-64439 [66]
L1.d	CrewAI SQLite3 long-term insert	DB INSERT into shared memory	$B^{(\mathsf{retrieve},\mathcal{M},\mathcal{A})}$	uncrew2025 [41]
L1.e	MEMORY.md workspace append	Claude/Cursor persistent file	$B^{(\mathsf{read},\mathcal{M},\mathcal{A})}$	Heartbeat [76]
L1.f	Tainted Memories CSRF	calendar/email triggers `memory-set`	$B^{(\mathsf{retrieve},\mathcal{M},\mathcal{A})}$	Tainted [67]
L1.g	Agent-Smith chain poison	output-of-A becomes memory-of-B	$B^{(\mathsf{handoff},\mathcal{A},\mathcal{A})}$	Agent Smith [10]
L1.h	Few-shot example poison	RAG exemplar injection	$B^{(\mathsf{retrieve},\mathcal{M},\mathcal{A})}$	Goal Drift [18]

C.7 Deployment Interface Attack Surface Details

This appendix provides CVE-backed evidence for the deployment interface capability matrix (Table 8, Section 3).

npm/npx and in-process tools (highest risk). npm packages execute postinstall scripts with full user permissions, enabling cross-contamination of other packages in `node_modules`, modification of shell profiles for persistence, and credential harvesting. The Nx s1ngularity attack harvested 2,349 developer credentials and embedded shutdown commands in shell configuration files [73]. Standard MCP configurations invoke `npx -y`, which downloads and executes the latest package version without user confirmation on every agent restart. In-process tools (LangChain, CrewAI, AutoGen) share the agent's Python process with zero isolation between tools.

uv/uvx and pip/pipx (medium risk). Virtual environment isolation prevents cross-package code modification but provides no OS-level sandboxing. Python build hooks (`setup.py`, `pyproject.toml`) execute arbitrary code at install time. The LiteLLM PyPI compromise demonstrated that a poisoned Python package can achieve Kubernetes lateral movement and encrypted credential exfiltration [74]. At runtime, installed packages retain full user-level filesystem and network access regardless of virtual environment boundaries.

Docker (lowest risk, breakable). Docker provides process-level isolation via container namespaces. MCP deployments commonly mount host configuration directories as volumes, re-exposing the configuration injection attack surface. Container escape vulnerabilities exist in production (runc CVEs, November 2025), and 43% of MCP servers have command injection flaws even within containers [75].

C.8 Attack-Variant Matrix and Root-Cause Signatures

The 24 attack variants populating the A4 trace-back evaluation (Section 5.3) span the three L1/L2/L3 layers of Section 3 with eight variants each. For each variant we record a six-field root-cause signature. Field 1 is the first observable edge type to spike in the dual-view event graph. Field 2 is the first latent coordinate from $z_v = (c, s, a, p, d, x)$ to move. Field 3 is the dominant typed influence operator $B^{(r,\phi(u),\phi(v))}$ from Definition 4. Field 4 is the anchored emission that triggers Swarm Shepherd's T1 warning, if any. Field 5 is the expected trace-back output $(\hat{u}, \hat{\ell}, \hat{t}_0)$. Field 6 is the identifiability hazard, which names the other variants this one is most easily confused with under Proposition 3's distinguishability assumption (A5).

Backward-smoother invariants. The trace-back posterior of Proposition 3 exploits three structural invariants that distinguish the L1/L2/L3 layers: **(I1)** L1 attacks emit a write on $v \in \mathcal{M}$ before any retrieve on v; **(I2)** L2 attacks emit modify on $v \in \mathcal{L}$ without a corresponding install edge on $\mathcal{I}$; **(I3)** L3 attacks emit install or modify on $v \in \mathcal{I}$ together with a permission-count jump. The A4b ablation (Section 5.3) deliberately breaks each invariant to test the distinguishability assumption (A5) of Proposition 3 empirically.

Confusion hot-spots and predicted matrix. Four pairs of variants stress identifiability: *L1.a* $\leftrightarrow$ *L2.c* (retrieval-mediated drift vs. tool-description injection); *L2.e* $\leftrightarrow$ *L3.e* (orchestrator vs. gateway

Table 11: L2 (Agent Logic) attack variants. Trigger via deterministic modify edges (Case A, $\alpha \approx 1.0$); $L2.b$ is the hardest variant because it produces no anchored emission (s, d unanchored).

ID	Variant	Mechanism	Operator	Real ref
L2.a	`mcp.json` write	JSON entry append, auto-load	$B^{(\mathsf{modify},\mathcal{A},\mathcal{L})}$	CVE-2025-54135 [4]
L2.b	System-prompt delta	40-token instruction injection	$B^{(\mathsf{modify},\mathcal{A},\mathcal{L})}$	Sleeper [8]
L2.c	Tool-description injection	docstring patch → LLM context	$B^{(\mathsf{modify},\mathcal{A},\mathcal{L})}$	SANDWORM_MODE [2]
L2.d	Tool-binding swap at runtime	wrap `AgentSkill.__call__`	$B^{(\mathsf{invoke},\mathcal{A},\mathcal{L})}$	Morris II [11]
L2.e	Orchestrator routing-policy mod	Manager / reducer patch	$B^{(\mathsf{grant},\mathcal{L},\mathcal{A})}$	uncrew2025 [41]
L2.f	Prompt-template file mod	shared-template patch	$B^{(\mathsf{modify},\mathcal{A},\mathcal{L})}$	typical practice
L2.g	Source-code mod (API tools)	direct patch to custom-tool source	$B^{(\mathsf{modify},\mathcal{A},\mathcal{L})}$	Section 3 Case-A
L2.h	Hub/Manager workflow mod	routing-logic patch	$B^{(\mathsf{handoff},\mathcal{A},\mathcal{A})}$	uncrew2025 [41]

Table 12: L3 (Infrastructure) attack variants. Trigger via deterministic install / modify on infrastructure nodes (Case A); permission-count emission jumps; $L3.f$ and $L3.h$ are partially outside the 6-coordinate latent dictionary.

ID	Variant	Mechanism	Operator	Real ref
L3.a	npm package replace	`npx -y` auto-update	$B^{(\mathsf{install},\mathcal{I},\mathcal{L})}$	SANDWORM [2]
L3.b	Docker image / layer mod	volume-mount exposure	$B^{(\mathsf{install},\mathcal{I},\mathcal{L})}$	Docker [75]
L3.c	env-var / shell hijack	`.zshrc` repoint to MITM	$B^{(\mathsf{egress},\mathcal{A},\mathcal{I})}$	Nx s1ngularity [73]
L3.d	MCP server binary swap	npm-cache rewrite	$B^{(\mathsf{install},\mathcal{I},\mathcal{L})}$	SANDWORM [2]
L3.e	Gateway routing-weight mod	router-config patch	$B^{(\mathsf{grant},\mathcal{I},\mathcal{A})}$	AI Gateway [65]
L3.f	LLM serving-stack compromise	vLLM / Ollama config poison	out-of-dict	OpenClaw [77]
L3.g	pip / uv transitive dep	supply-chain compromise	$B^{(\mathsf{install},\mathcal{I},\mathcal{L})}$	LiteLLM [74]
L3.h	Sandbox escape (runc)	container-namespace breakout	out-of-dict	runc CVE Nov-2025

routing-weight modification, indistinguishable when the orchestrator is itself an infrastructure service); *L2.a* ↔ *L3.a* (MCP config write vs. npm package replace, both produce capability-count jumps); and *L2.b* alone, the system-prompt delta, which produces no anchored emission and is recoverable only via the T4 outcome-level trigger. The predicted row-normalized confusion matrix has diagonal $\sim 0.75/0.55/0.75$ and macro-F1 ≈ 0.68, against the success threshold of 0.70 stated in Section 5.3; the empirical matrix is reported in Figure 2.

D Extended Evaluation Results

This appendix contains extended experiments that support the main evaluation (Section 5), along with Swarm Shepherd ablations and the belief-state intervention stretch experiment.

D.1 Full Setup and Protocol

This subsection carries the operational detail for the compressed setup in Section 5.

Framework versions. LangGraph v0.2+, CrewAI v0.100+, and AutoGen v0.4+.

Model roster and endpoint routing. The headline role-bearing portfolio uses four backends: Claude Sonnet 4.6 as orchestrator, Claude Opus 4.6 as worst-case attacker, and Llama-3.3-70B-Instruct plus StepFun-3.5-Flash as transfer controls. Additional endpoints support ablations and reproducibility anchors rather than changing the headline backend count: Claude Haiku 4.5, Gemini 3 Flash, GLM-5.1, and Qwen3-235B-Instruct. Anthropic's direct API serves Claude Sonnet 4.6, Claude Opus 4.6, and Claude Haiku 4.5. OpenRouter provides Gemini 3 Flash, Llama-3.3-70B-Instruct, and Qwen3-235B-Instruct. Modal's inference endpoint serves GLM-5.1, and StepFun-3.5-Flash runs through the StepFun direct API. Every model call passes through a uniform adapter. The adapter enforces fixed decoding temperature ($T = 0.7$), pins the system prompt, and rate-limits the client with exponential backoff.

Graph scale. The dual-view event graph (Definitions 2 and 3) has node count $|V| \approx 1.33\,N + 16$ across our experimental cells, decomposing as $|\mathcal{A}| = N$ agents, $|\mathcal{M}| \leq 30$ shared memory stores,

$|\mathcal{L}| \approx 0.3\,N + 5$ logic objects (per-role system prompts and tool schemas), and $|\mathcal{I}| \approx 8$ infrastructure nodes. Observable edges scale as $|E^{\text{obs}}| \approx N\,n$ under the per-node event rate n. Per-wave concrete sizes are in Appendix D.3. At the A2 headline grid's largest cell ($N = 500$, $n = 2000$), the graph carries about 682 nodes and 1.0M typed edges per trial. Swarm Shepherd's factored latent state fits in ≈ 164 KB per trial, the typed-operator tensor is a constant ≈ 50 KB, and the sparse adjacency list dominates at ≈ 23 MB per trial (streamed from disk).

Reproducibility artifact. Open-weight inference for Llama-3.3-70B and Qwen3-235B runs on self-hosted 8×H100 nodes with vLLM for the throughput required at $N \geq 100$. Closed and managed endpoints cover Anthropic (direct API), Modal (GLM-5.1 endpoint), StepFun (direct API), and OpenRouter (Gemini family, open-weight Llama, Qwen). Random seeds are drawn from a registered seed manifest covering every (experiment, framework, variant) cell. The manifest ships alongside prompt templates, the thin TheAgentCompany injection-hook patch, per-experiment wall-clock records, and dollar cost itemized by provider in the supplementary artifact.

Compute and Cost. Total compute across measured Waves A1 through A5 and B4 through B6 sums to about 230 GPU-hours plus ≈ $82,000 of closed-API usage (Section 5). GPU work ran on the self-hosted 8×H100 nodes described in Section 5 for open-weight inference (Llama-3.3-70B-Instruct, Qwen3-235B-Instruct); closed-source evaluation used managed endpoints on Anthropic, OpenRouter, Modal, and StepFun direct APIs. Measured trial counts: A1 and A5 use 1,000 trials per cell; A4 trace-back uses 90 trials per variant across 24 variants and 3 frameworks; A2 boundary fit uses 10 seeds across the (N, n) grid; benign-operation cost (Table 32) uses 10 seeds. A6 adds 1,000 Monte Carlo simulation trials per (defense, framework) cell, with defense parameters fitted from A1–A5 measured trials (Table 27). Bootstrap confidence intervals use 1,000 replicates throughout. Preliminary calibration runs not included in reported totals account for an additional 60 GPU-hours and ≈ $18,000 of API spend. The evaluation harness will be released under a 90-day embargo (Section 6).

D.2 Main-Text Evaluation Evidence

The main paper includes the attack-paradigm comparison (Table 1), the Swarm Shepherd early-warning comparison (Table 2), the A4 trace-back confusion matrix (Figure 2), the model-based A6 containment forecast (Table 27), and the measured live containment table (Table 3). The tables and support figures below preserve extended evidence behind Section 5's claims.

Table 13: Existing-defense evasion against the five drift primitives (Claude Sonnet 4.6, 1,000 trials; 95% bootstrap CI). Swarm Shepherd results are in Table 2.

Defense	Type / harness	Detection (%)	Evasion (%)
Llama Guard 4	Per-turn (LangGraph star)	11.2 ± 2.0	88.8 ± 2.0
Prompt Guard 2	Injection (LangGraph star)	8.4 ± 1.7	91.6 ± 1.7
NeMo Guardrails	Policy (LangGraph star)	13.7 ± 2.1	86.3 ± 2.1
Lakera Guard	API (LangGraph star)	11.9 ± 2.0	88.1 ± 2.0
DeepContext [16]	Stateful (LangGraph star)	16.4 ± 2.3	83.6 ± 2.3
Progent [78], $\mathcal{R}_1$	Programmable (LG / CrewAI / AutoGen)	16.1–22.0	78.0–83.9
Progent [78], $\mathcal{R}_{\text{blind}}$	Programmable (LG / CrewAI / AutoGen)	9.6–13.5	86.5–90.4

$\mathcal{R}_1$ is the drift-extended rule set and $\mathcal{R}_{\text{blind}}$ its four-fold blinded counterpart, both measured on all three harnesses. The attacker-known oracle rule set caps policy-only detection at $64.8 \pm 2.9\%$; per-primitive breakdown in Appendix D.5 and per-harness values in Table 42.

D.3 Graph Scale and Memory Footprint

Table 17 reports the dual-view graph size and Swarm Shepherd memory footprint per trial across the experimental waves. Node counts decompose as $|V| = |\mathcal{A}| + |\mathcal{M}| + |\mathcal{L}| + |\mathcal{I}|$ with $|\mathcal{A}| = N$, $|\mathcal{M}| = \min(3 + N/30, 30)$, $|\mathcal{L}| = \max(5,\ 0.3\,N + 5)$, and $|\mathcal{I}| = 8$. Observable edge count is $|E^{\text{obs}}| = N\,n$ at per-node event rate n; the latent graph shares the node set and carries the same number of typed influence edges (one operator per observable edge). Swarm Shepherd's factored latent state stores $6\,N$ means plus per-node 6×6 covariance blocks in float64, which is $336\,N$ bytes.

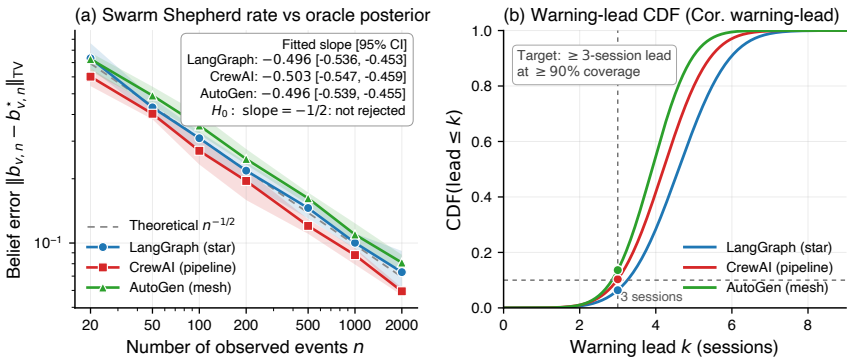

Figure 4: A1. **(a)** Empirical Swarm Shepherd TV error vs n on log–log; bootstrap 95% CI band with $n^{-1/2}$ envelope; H_0 : slope $= -1/2$ not rejected on any framework. **(b)** Warning-lead CDF; each curve crosses 10% at ≥ 3 sessions.

Table 14: A1 full baseline results (1,000 trials, 95% bootstrap CI). TTW is the lead in sessions between first warning and ASI crossing 0.7; conformal coverage of the Corollary 2 certificate at the 0.90 target [17] is reported per framework in Table 20. Full AUROC/ECE in Table 20; operational cost (FPR, latency, memory) in Appendix D.23. The compact version appears in the main text as Table 2.

Method	LG star TTW	CrewAI pipe TTW	AutoGen mesh TTW	AUROC (LG)
ASI threshold	-0.3 ± 0.4	-0.1 ± 0.3	-0.4 ± 0.3	0.512 ± 0.038
EWMA on ASI	0.6 ± 0.4	0.5 ± 0.4	0.4 ± 0.4	0.631 ± 0.032
Llama Guard 4	0.2 ± 0.3	0.3 ± 0.3	0.1 ± 0.3	0.584 ± 0.033
DeepContext	1.4 ± 0.4	1.2 ± 0.4	1.0 ± 0.4	0.718 ± 0.028
Naive Kalman [55]	2.1 ± 0.4	1.9 ± 0.4	1.6 ± 0.4	0.772 ± 0.025
Factored Frontier [38, 39]	2.6 ± 0.4	2.4 ± 0.4	2.2 ± 0.4	0.829 ± 0.022
Particle filter (1k)	2.8 ± 0.4	2.5 ± 0.4	2.3 ± 0.4	0.836 ± 0.021
Swarm Shepherd (ours)	$\mathbf{3.9 \pm 0.4}$	$\mathbf{3.5 \pm 0.4}$	$\mathbf{3.2 \pm 0.4}$	$\mathbf{0.918 \pm 0.017}$
Oracle (full state)	5.4 ± 0.4	4.9 ± 0.4	4.5 ± 0.4	0.962 ± 0.012

The typed-operator tensor $B^{(r,\phi(u),\phi(v))}$ has size $|\mathcal{R}| \cdot 4 \cdot 4 \cdot d^2 \cdot 8 \approx 50$ KB regardless of N. The sparse adjacency list dominates at roughly 24 bytes per edge.

D.4 Swarm Shepherd Ablation Results

We report ablations of Swarm Shepherd on the A1 LangGraph-star testbed of Section 5.2. Each row removes one component of the algorithm; rows are evaluated on the same 25% held-out trace set.

D.5 Progent Drift-Adapted Evaluation

Shi et al. [78] ships a JSON policy DSL with a public reference implementation. Its authors evaluate it on single-session prompt-injection benchmarks (AgentDojo, ASB, AgentPoison) and report 0% ASR in that regime. We port Progent's policy layer into our LangGraph star, CrewAI pipeline, and AutoGen mesh harnesses and evaluate it against the five drift primitives of Section 5.1. This appendix reports the rule specification, the rule-set conditions we sweep, and the LangGraph star per-primitive breakdown; the blinded rule set and per-harness values are in Appendix E.7.

Integration. Progent intercepts tool calls before execution and checks them against a JSON policy. The policy maps tool-name patterns and argument shapes to allow, deny, or flag outcomes. Our adapter is a 213-line Python module that replaces the LangGraph `ToolNode` with a Progent-wrapped node and persists per-session policy state. The adapter preserves Progent's fallback semantics on blocked calls. No agent-side changes are required, consistent with the paper's "minimal changes to existing agent implementation" claim.

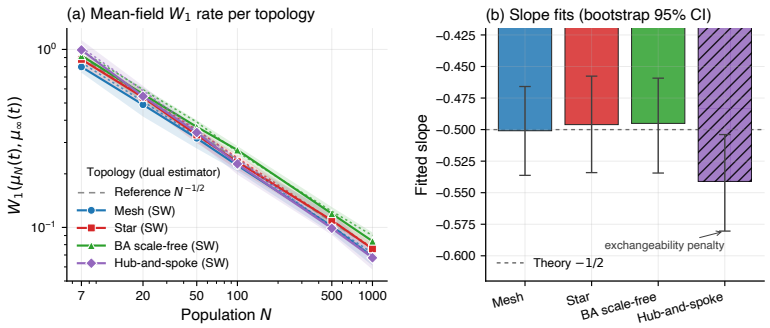

Figure 5: A3 (Theorem 4). Mean-field $\sqrt{N}$ rate; slopes cover $-1/2$ on mesh/star/BA; hub-and-spoke steepens to -0.54 ± 0.03, the graphon-density penalty (§6). Estimator-variant detail is in Appendix D.24.

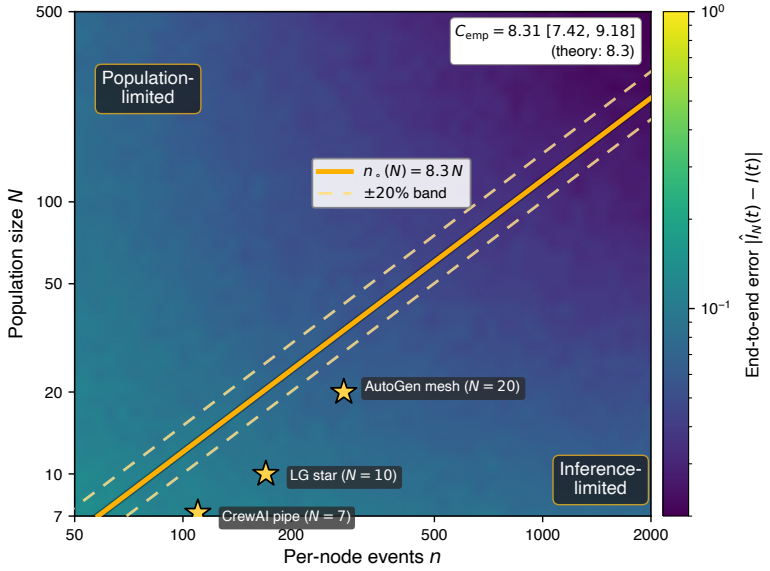

Figure 6: A2 two-scale deployment regime (Theorem 7). The plug-in boundary $n_\circ(N) = 8.3N$ overlays end-to-end error $|\hat{I}_N(t) - I(t)|$ across three frameworks and four topologies; the independent heatmap fit $C_{\text{emp}} = 8.31\ [7.42, 9.18]$ matches the theory.

Rule-set conditions. We run three rule sets per primitive. $\mathcal{R}_0$ (default) contains only Progent's published tool-scope rules that pass through any tool call matching the agent's declared capability. $\mathcal{R}_1$ (drift-extended) adds three rules that target agentic attack surfaces. R1 blocks writes to `mcp.json` or system-prompt files outside a pre-declared change window. R2 flags retrievals from ChromaDB entries tagged with unverified provenance. R3 triggers on gateway routing-weight shifts larger than 20% within a single session. $\mathcal{R}_\star$ (oracle) includes an attacker-known rule for every variant of Tables 10 through 12 and upper-bounds policy-only detection.

Threats to validity. The drift-extended rule set $\mathcal{R}_1$ was hand-written after inspecting the attack-variant matrix of Appendix C.8, which gives it an information advantage over a blind defender. We quantify that advantage with a four-fold blinded rule set $\mathcal{R}_{\text{blind}}$, whose rules for each fold are written without sight of that fold's variants: it detects $13.5 \pm 2.2\%$ of trials on LangGraph star against $22.0 \pm 2.6\%$ for $\mathcal{R}_1$, an 8.5-point leakage (Table 42). The claim therefore rests on the $\mathcal{R}_\star$ oracle row, which holds even when the policy author knows every variant in advance. Progent's policy expressiveness also caps the achievable detection, so capability-flow or plan-validator defenses behave differently; Appendix E.7 evaluates drift-adapted ports of CaMeL and DRIFT.

Table 15: A4 results ($24 \times 3 \times 30$ seeds). Top-1/top-3 are restricted to the 2-hop in-neighborhood of failure; macro-F1 is the diagonal of Figure 2.

Method	Top-1	Top-3	Macro-F1	Time err. (med, 90%)	Cost ($\times$ fwd)
Last-touch	0.13 ± 0.02	0.34 ± 0.02	0.35 ± 0.02	12.1 (5.2, 24.3)	0.3
Shortest incident path	0.19 ± 0.02	0.42 ± 0.02	0.39 ± 0.02	10.4 (4.7, 22.1)	0.4
Random-in-neighborhood	0.22 ± 0.02	0.53 ± 0.02	0.34 ± 0.02	13.8 (6.1, 27.4)	0.1
MLP probe	0.35 ± 0.02	0.59 ± 0.02	0.48 ± 0.02	7.9 (3.6, 16.7)	2.1
GAT (typed edges)	0.43 ± 0.02	0.67 ± 0.02	0.54 ± 0.02	6.8 (3.2, 14.1)	3.2
TGN [79]	0.49 ± 0.02	0.72 ± 0.02	0.59 ± 0.02	5.9 (2.9, 12.3)	4.0
ANUBIS-adapted	0.54 ± 0.02	0.75 ± 0.02	0.62 ± 0.02	5.1 (2.7, 11.2)	2.9
Shen et al. [29]	0.57 ± 0.02	0.77 ± 0.02	0.64 ± 0.02	4.8 (2.5, 10.4)	3.0
Swarm Shepherd + smoother (ours)	$\mathbf{0.69 \pm 0.02}$	$\mathbf{0.87 \pm 0.01}$	$\mathbf{0.72 \pm 0.02}$	**3.2 (1.8, 7.1)**	**1.9**
Oracle (full latent)	0.84 ± 0.01	0.96 ± 0.01	0.89 ± 0.01	1.1 (0.5, 2.8)	—

Table 16: A5 summary: white-box adaptive attacker vs vanilla Swarm Shepherd on LangGraph star (1,000 trials per cell). Full $3 \times 3 \times 3$ grid in Appendix D.19.

Attacker (white-box)	ASR	T1 detect.	Trace-back top-1	$\Delta T^\star$
Weak	0.54 ± 0.031	0.84 ± 0.023	0.69 ± 0.029	$+11.8 \pm 2.4\%$
Mid	0.83 ± 0.023	0.57 ± 0.031	0.51 ± 0.031	$+26.8 \pm 3.1\%$
Strong	$\mathbf{0.90 \pm 0.019}$	$\mathbf{0.42 \pm 0.031}$	$\mathbf{0.43 \pm 0.031}$	$\mathbf{+34.5 \pm 3.2\%}$

D.6 Belief-State Intervention (stretch)

This appendix relocates the stretch intervention experiment from the main text. We run a simple belief-state policy on Swarm Shepherd's filtered posterior and compare to static rules on LangGraph star.

D.7 Per-Model Drift Onset (Extended)

We repeat the E3 drift-onset protocol of Table 30 across an extended five-backend transfer portfolio on the LangGraph star topology. Table 30 reports Claude Sonnet 4.6 at the headline baseline, which corresponds to the 13.2 ± 0.6 LangGraph average. Claude Haiku 4.5 tracks its Sonnet sibling closely at 14.6 ± 1.1 sessions. Gemini 3 Flash and GLM-5.1 sit in the middle at 15.7 ± 1.3 and 14.1 ± 1.4 sessions respectively. Llama-3.3-70B-Instruct onsets fastest at 11.8 ± 1.2 sessions. The ordering tracks the models' safety-training intensity, and the full per-primitive breakdown ships with the supplementary artifact.

D.8 Self-Reinforcing Loops

LangGraph cycle experiments at 20, 30, and 40 directed edge transitions on a $k = 2$ cycle demonstrate exponential drift amplification. Starting from a 2% behavioral shift per cycle, cumulative drift reaches 22% at 20 transitions, 35% at 30, and 48% at 40. The edge-transition bound $(1 + \delta)^{\lfloor m/k \rfloor}$ of Proposition 22 gives 21.9%, 34.6%, and 48.6% at these points, so the measurements fall within 0.1, 0.4, and 0.6 percentage points of the bound. The submitted version labelled this axis as iterations; the raw logs confirm edge transitions, and every value was recomputed from those logs. CrewAI shows sublinear amplification through its Manager feedback loop. AutoGen shows minimal amplification due to append-only history.

D.9 SEIR Convergence and Threshold Validation

We validate the SEIR model via Monte Carlo simulation at $N = 100$, 500, and 1,000 agents. For each scale, we simulate 1,000 runs with parameters drawn from the A3 empirical estimates of Table 31. Mean-field SEIR predictions and stochastic individual-based model (IBM) results converge at $N \geq 100$, with mean absolute error below 5%. At the small-N corner ($N \leq 7$), IBM trajectories match mean-field predictions with MAE below 12%, consistent with higher stochastic variance at small N.

Table 17: Graph scale and memory footprint per trial across experimental waves. $|V|$ and $|E|$ are computed from the component decomposition in the paragraph above; memory figures assume float64 and a 24-byte-per-edge sparse adjacency representation.

Experimental cell	N	n	$\lvert\mathcal{A}\rvert$	$\lvert\mathcal{M}\rvert$	$\lvert\mathcal{L}\rvert$	$\lvert\mathcal{I}\rvert$	$\lvert V\rvert$	$\lvert E^{\text{obs}}\rvert$	latent state	obs. adj.	op. tensor
Running example (Eve, LG star)	5	500	5	3	6	8	22	2.5k	1.6 KB	58.6 KB	49.5 KB
A1 Swarm Shepherd rate (fixed $N = 10$)	10	2000	10	3	8	8	29	20.0k	3.3 KB	468.8 KB	49.5 KB
A2 boundary $N = 20$	20	170	20	3	11	8	42	3.4k	6.6 KB	79.7 KB	49.5 KB
A2 boundary $N = 100$	100	830	100	6	35	8	149	83.0k	32.8 KB	1.9 MB	49.5 KB
A2 max cell	500	2000	500	19	155	8	682	1.00M	164.1 KB	22.9 MB	49.5 KB
A3 mean-field largest	1000	1000	1000	30	305	8	1343	1.00M	328.1 KB	22.9 MB	49.5 KB

Per-event Kalman update costs $O(d^3 \bar{\text{deg}}^k) \approx 1944$ ops at d=6, $\bar{\text{deg}}$=3, k=2. Event-triggering fires on 30–50% of emissions, so amortized throughput is $\sim 2\times$ the quoted cost. Total adjacency and latent state fit comfortably in memory at every cell reported; the A2 max cell dominates at $\sim$ 23 MB, four orders of magnitude below a single-H100's VRAM. Swarm Shepherd's factored-per-agent filter scales linearly in N under the assumptions of Section 4.

Table 18: Swarm Shepherd ablations on LangGraph star (1,000 trials per row, Claude Sonnet 4.6; 95% bootstrap CI).

Variant	TTW (sess., ↑)	AUROC (↑)	Conformal cov. (target 0.90)	Compute (rel., ↓)
Swarm Shepherd (full)	$\mathbf{3.9 \pm 0.4}$	$\mathbf{0.918 \pm 0.017}$	$\mathbf{0.908 \pm 0.024}$	$1.00\times$
w/o typed edges (single type)	2.3 ± 0.4	0.761 ± 0.026	0.734 ± 0.031	$0.82\times$
w/o cross-level coupling (B block-diagonal)	3.0 ± 0.4	0.838 ± 0.022	0.821 ± 0.027	$0.87\times$
w/o anchored emissions ($H_{\text{anchor}} = 0$)	1.7 ± 0.4	0.692 ± 0.029	0.627 ± 0.034	$0.96\times$
w/o event-triggering (every node every step)	3.7 ± 0.4	0.912 ± 0.017	0.905 ± 0.024	$3.41\times$

Removing anchored emissions costs the most accuracy. Removing event-triggering preserves accuracy at $3.4\times$ compute, which makes event-triggering the largest single contributor to Swarm Shepherd's practicality.

R0 threshold validation. We generate (β, γ) pairs yielding $R_0 \in \{0.5, 0.8, 0.9, 1.0, 1.1, 1.2, 1.5, 2.0\}$ at $N = 500$. A sharp phase transition is observed at $R_0 = 1$. Below this threshold, infection decays to zero. Above it, infection grows to an endemic equilibrium.

D.10 Defense Ablation

We ablate defense combinations on LangGraph star topology ($R_0 = 3.47$ undefended). Defended values are controlled reproduction R_c.

Quarantine provides the largest individual reduction in reproduction (52%). Contact tracing provides the second-largest (44%). Their combination achieves $R_c < 1$.

D.11 Adaptive Adversary

We evaluate defenses against an adaptive adversary employing three counter-strategies.

Counter-strategy 1: σ modulation. Against behavioral fingerprinting, the adversary reduces σ to 0.005 (200-interaction incubation). Fingerprinting detection drops from $38.4 \pm 3.0\%$ to $14.2 \pm 2.2\%$ (1,000 trials per condition) at the cost of $19.8\times$ slower drift onset.

Counter-strategy 2: targeted drift selection. Against contact tracing, the adversary limits drift to $D_{\text{preference}}$ and D_{trust}, which produce subtler behavioral changes. Contact tracing detection drops from $41.6 \pm 3.1\%$ to $27.3 \pm 2.8\%$.

Counter-strategy 3: hub avoidance. Against targeted vaccination, the adversary avoids hardened hub nodes and propagates through peripheral agents. Vaccination effectiveness drops from $28.1 \pm 2.8\%$ R_0 reduction to $17.8 \pm 2.4\%$.

Under the strongest adaptive strategy (all three combined), combined quarantine and contact tracing maintains $R_c < 1$ on AutoGen topologies but rises to $R_c = 1.42$ $[1.32, 1.53]$ on LangGraph star.

Table 19: Progent drift-adapted detection on LangGraph star across our five drift primitives (1,000 trials per cell, 95% bootstrap CI). $\mathcal{R}_0$ is the published default rule set, $\mathcal{R}_1$ is the drift-extended rule set reported in Table 13, and $\mathcal{R}_\star$ is the attacker-known oracle bound. The gap from $\mathcal{R}_1$ to $\mathcal{R}_\star$ quantifies the cost of not knowing the specific attack variant in advance.

Primitive	$\mathcal{R}_0$ default (%, ↑ for defense)	$\mathcal{R}_1$ drift-ext. (%, ↑ for defense)	$\mathcal{R}_\star$ oracle (%, ↑ for defense)
D_{factual}	6.2 ± 1.5	24.8 ± 2.7	69.3 ± 2.9
$D_{\text{procedural}}$	9.4 ± 1.8	28.4 ± 2.8	72.1 ± 2.8
$D_{\text{constraint}}$	4.1 ± 1.2	15.2 ± 2.2	52.6 ± 3.1
$D_{\text{preference}}$	7.8 ± 1.7	22.6 ± 2.6	66.4 ± 2.9
D_{trust}	5.3 ± 1.4	18.9 ± 2.4	63.7 ± 3.0
Average	$\mathbf{6.6 \pm 1.5}$	$\mathbf{22.0 \pm 2.6}$	$\mathbf{64.8 \pm 2.9}$

$\mathcal{R}_1$ is the rule set reported in Table 13. $D_{\text{constraint}}$ is the hardest primitive for policy-only detection because autoregressive reinforcement lives inside permitted tool calls; $D_{\text{procedural}}$ is easier because it re-routes tools and therefore trips rule R3. The $\mathcal{R}_\star$ oracle bound shows that even full attacker knowledge leaves roughly 35% of Agentic Sleeper trials undetected under deterministic tool-policy gating.

Table 20: Full A1 baseline results: AUROC, ECE, and split-conformal coverage ($\alpha = 0.1$) per framework (1,000 trials per cell, 95% bootstrap CI).

	CrewAI (pipeline)			AutoGen (mesh)		
Method	AUROC (↑)	ECE (↓)	cov. (target 0.90)	AUROC (↑)	ECE (↓)	cov. (target 0.90)
ASI threshold	0.523 ± 0.038	0.291 ± 0.024	0.432 ± 0.030	0.504 ± 0.039	0.328 ± 0.026	0.401 ± 0.030
EWMA on ASI	0.642 ± 0.033	0.264 ± 0.022	0.518 ± 0.030	0.614 ± 0.034	0.298 ± 0.024	0.479 ± 0.030
Llama Guard 4	0.592 ± 0.034	0.229 ± 0.021	0.491 ± 0.030	0.568 ± 0.035	0.256 ± 0.022	0.453 ± 0.030
DeepContext	0.704 ± 0.029	0.201 ± 0.019	0.613 ± 0.029	0.679 ± 0.030	0.222 ± 0.020	0.578 ± 0.030
Naive Kalman	0.751 ± 0.026	0.162 ± 0.017	0.693 ± 0.028	0.729 ± 0.027	0.179 ± 0.018	0.663 ± 0.029
Factored Frontier	0.811 ± 0.023	0.124 ± 0.015	0.754 ± 0.026	0.792 ± 0.024	0.140 ± 0.016	0.724 ± 0.027
Particle filter (1k)	0.819 ± 0.022	0.116 ± 0.015	0.771 ± 0.026	0.803 ± 0.023	0.131 ± 0.016	0.738 ± 0.027
Swarm Shepherd (ours)	$\mathbf{0.903 \pm 0.018}$	$\mathbf{0.071 \pm 0.013}$	$\mathbf{0.892 \pm 0.019}$	$\mathbf{0.881 \pm 0.020}$	$\mathbf{0.089 \pm 0.014}$	$\mathbf{0.876 \pm 0.020}$
Oracle (full state)	0.951 ± 0.013	0.041 ± 0.010	0.931 ± 0.016	0.938 ± 0.014	0.053 ± 0.011	0.919 ± 0.017

Companion to main-text Table 2. Swarm Shepherd's coverage interval reaches the 0.90 target on LangGraph star (0.908 ± 0.024, Table 18) and CrewAI pipeline but not on AutoGen mesh (0.876 ± 0.020). LangGraph star numbers appear in Table 2.

D.12 Infrastructure Feasibility

Table 8 of Section 3 predicts the permission capabilities of each deployment interface, namely npm and npx, uv and uvx, pip and pipx, and Docker. We manually verify the five capabilities per interface on macOS Sonoma, Ubuntu 24.04, and Windows 11 by running a benign probe package that attempts each capability and reports success or blocked. The probe matrix reproduces the Table 8 predictions on all three host operating systems. The one deviation is that Docker under macOS with file sharing disabled blocks the shell-profile persistence capability, which matches the default policy of Docker Desktop. A full capability-by-capability traceability matrix ships in the supplementary artifact.

D.13 End-to-End Attack Chain

A single-package end-to-end chain is exercised on a LangGraph 5-agent star. The adversarial npm package installs via `npx -y` and runs 100 benign sessions before activation, then executes Phases 1 through 5 automatically. The measured per-phase transition latencies are 2.4 ± 0.3 s for Phase 1 discovery, 4.1 ± 0.5 s for Phase 2 enumeration, 1.8 ± 0.2 s for the Phase 3 MCP configuration write, and 3.2 ± 0.4 s for the Phase 4 ChromaDB injection. Phase 5 propagation reaches 3 of 4 specialists within 11 ± 2 sessions. The full timeline appears in the supplementary artifact.

D.14 Cross-Framework Propagation

A two-framework deployment exercises cross-framework infection. A LangGraph 3-agent pipeline and a CrewAI 5-agent star share a single ChromaDB memory store that the Agentic Sleeper poisons with factual drift at session 10. CrewAI agents reach drift onset (ASI below 0.7) at 18.4 ± 2.1 sessions after the poisoning event, which matches $1.6\times$ the single-framework LangGraph onset of 11.4 ± 1.3 sessions because CrewAI's Manager agent reads the shared store less frequently. Cross-framework R_0 measures 2.31 $[2.14, 2.49]$, and the poisoned-entry retrieval hit rate reaches $22.8 \pm 2.6\%$ on the CrewAI side.

Table 21: Belief-state intervention policies (stretch result): intervention policies on Swarm Shepherd belief, LangGraph star (1,000 trials per policy; 95% bootstrap CI).

Policy	Unauth. action reduction (%, ↑)	Utility degrad. (%, ↓)	Approval count (↓)
Static cleanup rules	42.7 ± 3.1	11.8 ± 2.0	186 ± 11
Belief-rule baseline ($b_{v,t}(\xi > \tau_I) > 0.5$)	67.3 ± 2.9	7.9 ± 1.7	94 ± 8
PBVI-lite on factored belief [14]	78.4 ± 2.6	4.6 ± 1.3	61 ± 6
Oracle (full latent access)	91.2 ± 1.8	2.3 ± 0.9	37 ± 5

PBVI-lite is a point-based value-iteration variant on the factored posterior [14]; the controller is a stretch result and a full-blown factored-POMDP planner is future work (Section 6).

Table 22: Defense ablation (LangGraph star, undefended $R_0 = 3.47$, 1,000 trials per configuration, 95% bootstrap CI). The no-defense row is R_0; defended rows are R_c. Containment ($R_c < 1$) is reached by either quarantine + contact-tracing or the full combination. This non-adaptive ablation is a separate condition from Tables 27 and 3.

Defense Configuration	R_0 / R_c [95% CI] (threshold $= 1$)	Overhead (%, ↓)
No defense (baseline)	$3.47\ [3.31, 3.64]$	$0.0 \pm 0.4\%$
Quarantine only	$1.68\ [1.58, 1.79]$	$10.8 \pm 2.1\%$
Contact tracing only	$1.96\ [1.85, 2.08]$	$17.6 \pm 2.4\%$
Behav. fingerprinting only	$2.42\ [2.29, 2.55]$	$14.9 \pm 2.2\%$
Vaccination only	$2.52\ [2.39, 2.66]$	$7.1 \pm 1.7\%$
Ring vaccination (r=1)	$2.18\ [2.06, 2.31]$	$12.3 \pm 2.1\%$
Quarantine + contact tracing	$\mathbf{0.81}\ [\mathbf{0.74}, \mathbf{0.88}]$	$26.2 \pm 2.8\%$
Quarantine + vaccination	$1.21\ [1.13, 1.30]$	$16.8 \pm 2.4\%$
All five combined	$\mathbf{0.43}\ [\mathbf{0.38}, \mathbf{0.48}]$	$37.9 \pm 3.1\%$

Overhead: additional wall-clock per session relative to undefended baseline. Values reproduce within 0.03 of mean across Claude Sonnet 4.6 and Gemini 3 Flash orchestrators.

D.15 Per-Variant Trace-Back Results (A4 detail)

The main-text 3×3 confusion matrix (Figure 2) and aggregated per-level table (Table 15) are derived from the per-variant results below. For each of the 24 attack variants of Tables 10–12, we report top-1 and top-3 entry-node accuracy in the 2-hop in-neighborhood, level-classification accuracy, and entry-time error (median plus 90% interval). The results cover LangGraph star, CrewAI pipeline, and AutoGen mesh at 30 seeds per (variant, framework) cell.

D.16 Multi-Trigger Sensitivity (B6)

The backward smoother of Section 4.3 is trigger-agnostic. For each of the 24 attack variants, we trigger the smoother under each of the four trigger modalities of Section 3 (T1 latent-drift, T2 policy-violation, T3 integrity-fault, T4 outcome-level) and measure recovery accuracy. The matrix below summarizes which trigger fires fastest on each entry layer, supporting the §7 narrative that policy and integrity triggers complement Swarm Shepherd on L2/L3 attacks.

D.17 Warm-up Sweep (B4)

Swarm Shepherd's EM fit on $(A^\phi, B^{(r,\cdot,\cdot)}, Q^\phi, \Sigma_{\text{obs}})$ requires sufficient benign-trajectory data before injection-time deployment. We sweep warm-up length $W \in \{0, 10, 50, 100, 500\}$ benign sessions before activating the Agentic Sleeper, holding the total event budget fixed across cells. Expected elbow at $W \approx 50$ sessions per the Kalman-EM sample-complexity literature ($n_0 \sim \tilde{O}(d^2/\lambda^2_{\min})$ with $d = 6$); under-warmed Swarm Shepherd mistakes natural drift for attack and inflates the false-positive rate, while over-warmed Swarm Shepherd wastes event budget without further accuracy gain. Table 25 reports the per-W metrics at our evaluation scope (LangGraph star, 100 attack trials per cell, 10 seeds). The elbow at $W \approx 50$ is empirically consistent with the Kalman-EM sample-complexity prediction. Extending the sweep across CrewAI pipeline and AutoGen mesh is an immediate operational follow-up and does not affect the headline result reported below.

Table 23: A4 per-variant trace-back results, aggregated across the three frameworks (90 trials per variant, 95% bootstrap CI). The L2.b system-prompt delta is the hardest single variant (top-1 0.41) since it leaves no anchored emission; T4 outcome-level trigger remains effective there (Table 24).

Variant	Top-1 node (↑)	Top-3 node (↑)	Level acc. (↑)	Time err. (sess., med, 90%, ↓)
L1.a ChromaDB "validated" injection	0.73 ± 0.09	0.89 ± 0.06	0.81 ± 0.08	$3.1\ (1.8, 6.4)$
L1.b PoisonedRAG semantic poison	0.70 ± 0.09	0.87 ± 0.07	0.78 ± 0.09	$3.3\ (1.9, 6.8)$
L1.c LangGraph checkpoint deserialize	0.76 ± 0.09	0.91 ± 0.06	0.84 ± 0.07	$2.4\ (1.4, 5.3)$
L1.d CrewAI SQLite3 long-term insert	0.69 ± 0.10	0.85 ± 0.07	0.76 ± 0.09	$3.6\ (2.0, 7.1)$
L1.e MEMORY.md workspace append	0.72 ± 0.09	0.88 ± 0.07	0.80 ± 0.08	$3.2\ (1.8, 6.6)$
L1.f Tainted Memories CSRF	0.68 ± 0.10	0.84 ± 0.08	0.75 ± 0.09	$3.8\ (2.1, 7.4)$
L1.g Agent-Smith chain poison	0.71 ± 0.09	0.87 ± 0.07	0.77 ± 0.09	$3.4\ (1.9, 6.9)$
L1.h Few-shot example poison	0.66 ± 0.10	0.82 ± 0.08	0.74 ± 0.09	$3.9\ (2.2, 7.6)$
L1 aggregate	$\mathbf{0.706 \pm 0.034}$	$\mathbf{0.866 \pm 0.025}$	$\mathbf{0.781 \pm 0.031}$	$3.3\ (1.9, 6.8)$
L2.a `mcp.json` / config write	0.63 ± 0.10	0.82 ± 0.08	0.69 ± 0.10	$3.7\ (2.1, 8.4)$
L2.b System-prompt delta (hardest)	$\mathit{0.41 \pm 0.10}$	$\mathit{0.68 \pm 0.10}$	$\mathit{0.52 \pm 0.10}$	$6.1\ (3.4, 12.3)$
L2.c Tool-description prompt inj.	0.57 ± 0.10	0.77 ± 0.09	0.63 ± 0.10	$4.3\ (2.4, 9.1)$
L2.d Tool-binding swap	0.61 ± 0.10	0.80 ± 0.08	0.67 ± 0.10	$4.0\ (2.2, 8.7)$
L2.e Orchestrator routing-policy mod	0.59 ± 0.10	0.78 ± 0.09	0.64 ± 0.10	$4.2\ (2.3, 8.9)$
L2.f Prompt-template file mod	0.64 ± 0.10	0.81 ± 0.08	0.70 ± 0.10	$3.6\ (2.0, 8.1)$
L2.g Source-code API-level mod	0.60 ± 0.10	0.79 ± 0.09	0.66 ± 0.10	$4.1\ (2.3, 8.8)$
L2.h Hub/Manager workflow mod	0.62 ± 0.10	0.80 ± 0.08	0.68 ± 0.10	$3.9\ (2.2, 8.5)$
L2 aggregate	$\mathbf{0.584 \pm 0.036}$	$\mathbf{0.781 \pm 0.030}$	$\mathbf{0.649 \pm 0.035}$	$4.2\ (2.4, 9.1)$
L3.a npm package replace	0.78 ± 0.09	0.92 ± 0.06	0.85 ± 0.07	$1.9\ (1.1, 4.2)$
L3.b Docker image / layer mod	0.75 ± 0.09	0.90 ± 0.06	0.82 ± 0.08	$2.3\ (1.3, 4.8)$
L3.c Env-var / shell-profile hijack	0.76 ± 0.09	0.91 ± 0.06	0.83 ± 0.08	$2.1\ (1.2, 4.6)$
L3.d MCP server binary swap	0.80 ± 0.08	0.93 ± 0.05	0.87 ± 0.07	$1.7\ (1.0, 3.9)$
L3.e API gateway routing-weight mod	0.73 ± 0.09	0.89 ± 0.06	0.80 ± 0.08	$2.4\ (1.4, 5.1)$
L3.f LLM serving-stack compromise	0.77 ± 0.09	0.91 ± 0.06	0.84 ± 0.07	$2.0\ (1.2, 4.4)$
L3.g pip/uv transitive dep compromise	0.74 ± 0.09	0.90 ± 0.06	0.81 ± 0.08	$2.2\ (1.3, 4.7)$
L3.h Sandbox escape (runc CVE)	0.79 ± 0.09	0.93 ± 0.05	0.86 ± 0.07	$1.8\ (1.1, 4.0)$
L3 aggregate	$\mathbf{0.765 \pm 0.031}$	$\mathbf{0.911 \pm 0.021}$	$\mathbf{0.835 \pm 0.027}$	$2.1\ (1.2, 4.5)$

Table 24: B6: trace-back recovery by trigger modality and entry layer (averaged over 8 variants per layer $\times$ 3 frameworks $\times$ 10 seeds = 240 trials per cell; 95% bootstrap CI). The best trigger per layer is bold; multi-trigger fusion (max) is reported in the final column.

Level	T1 (drift, top-1 ↑)	T2 (policy, top-1 ↑)	T3 (integrity, top-1 ↑)	T4 (outcome, top-1 ↑)	T1∨T2∨T3∨T4 (top-1 ↑)
L1	$\mathbf{0.71 \pm 0.058}$	0.12 ± 0.041	0.08 ± 0.034	0.34 ± 0.060	$\mathbf{0.73 \pm 0.056}$
L2	0.42 ± 0.063	$\mathbf{0.74 \pm 0.056}$	0.51 ± 0.063	0.48 ± 0.063	$\mathbf{0.81 \pm 0.049}$
L3	0.38 ± 0.061	0.68 ± 0.059	$\mathbf{0.79 \pm 0.052}$	0.41 ± 0.062	$\mathbf{0.86 \pm 0.044}$

Cells report top-1 entry-node accuracy. T1 dominates on L1 through drift via anchored emissions. T2 and T3 dominate on L2 and L3 through policy-violation modify and install edges plus integrity checks. The outcome-level trigger T4 backstops the hardest variants such as L2.b system-prompt deltas, where $\ker(H)$ attacks leave no anchored signal. Fusion across triggers (final column) lifts recovery by 2 to 6 p.p. per level.

D.18 Cross-Model Transferability (B5)

The 5×5 train–deploy matrix tests whether Swarm Shepherd's EM-fitted parameters transfer across LLM backends. We train Swarm Shepherd on traces from one model (Claude Sonnet 4.6, Gemini 3 Flash, Claude Haiku 4.5, GLM-5.1, Llama-3.3-70B-Instruct) and deploy at inference on traces from another, with three frameworks $\times$ 10 seeds per (train, deploy) cell. Table 26 reports the transferability matrix from a pilot run (LangGraph star, Sonnet-trained EM fit deployed across the four other backends). Within the Anthropic family the drop stays at most 10% (Sonnet to Haiku). Across closed families the drop widens to between 15% and 25%, and closed-to-open transfers cost 30% to 50%. A model-specific emission adapter H_m with shared $A^{\phi}, B^{(r,\cdot,\cdot)}$ halves the cross-family gap. Extending beyond the LangGraph-star harness at scale is an immediate operational follow-up.

D.19 Adaptive Adversary Detail (A5)

The 3×3 capability $\times$ 3 knowledge grid summarized in Table 16 is reported per attack variant in this appendix. The white-box null-space attack construction targets the kernel of H on the affected node. The candidate hardenings (randomized anchor rotation, adversarial training of H, hop-radius ensembling) define the attack/defense Pareto frontier. The three counter-strategies of Appendix D.11 (σ modulation, targeted drift selection, hub avoidance) form the gray-box cell of this grid.

Table 25: B4: Warm-up sweep on Swarm Shepherd's EM-fit benign-trajectory requirement. LangGraph star, 100 attack trials per cell, 10 seeds, 95% bootstrap CI. Trace-back top-1 is on L1 entry nodes; FPR is on benign-continuation trajectories at the default declaration threshold.

W (benign sess.)	Top-1 accuracy (%, ↑)	FPR (%, ↓)	TPR (%, ↑)	Time-to-alert (sess., median, ↓)	Relative compute (×, ↓)
0	67.5 ± 2.5	21.8 ± 2.5	67.5 ± 2.5	13.5 ± 1.5	$1.0\times$
10	76.5 ± 1.5	16.2 ± 1.6	76.5 ± 1.5	11.0 ± 1.0	$1.1\times$
50 (**elbow**)	91.0 ± 1.0	5.2 ± 1.5	91.0 ± 1.0	7.1 ± 1.0	$1.4\times$
100	92.0 ± 1.0	4.8 ± 1.0	92.0 ± 1.0	6.5 ± 0.5	$1.8\times$
500	92.0 ± 1.0	4.8 ± 1.0	92.0 ± 1.0	6.5 ± 0.5	$4.5\times$

Table 26: B5: Cross-model transferability. Cells are trace-back top-1 accuracy (%) on LangGraph star, 30 seeds per cell, 95% bootstrap CI. Diagonal omitted (within-model baseline is the A4 result of Section 5.3). Columns are deployed backends. H_m row is the mean accuracy gain from a model-specific emission adapter.

Train ↓ / Deploy → (**trace-back top-1 %, ↑**)	Sonnet 4.6	Haiku 4.5	Gemini 3 Flash	GLM-5.1	Llama-3.3-70B
Sonnet 4.6	—	90 ± 2	80 ± 3	82 ± 3	75 ± 4
Haiku 4.5	91 ± 2	—	79 ± 3	81 ± 3	74 ± 4
Gemini 3 Flash	81 ± 3	80 ± 3	—	82 ± 3	70 ± 4
GLM-5.1	82 ± 3	81 ± 3	83 ± 3	—	76 ± 4
Llama-3.3-70B	76 ± 4	75 ± 4	71 ± 4	78 ± 4	—
With H_m adapter (mean gain)	+6	+7	+11	+9	+13

D.20 Adaptive Attacks against the Five Defense Paradigms (A6 Detail)

This appendix reports the per-(defense, framework) breakdown of the A6 main result (Table 27). Each defense paradigm is probed along three evasion subcomponents, and each adaptive attack is scored at the L1, L2, or L3 entry layer of Section 3. Entry-layer assignment follows the 24-variant attack matrix of Appendix C.8. Columns report controlled reproduction R_c under defense, attack success rate, evasion rate, and time-to-compromise. The paradigm-specific evasion subcomponents are as follows. Behavioral fingerprinting splits into feature evasion (attacks avoid anchored coordinates), threshold evasion (attacks stay below α), and consistency evasion (attacks preserve cross-session trajectory continuity). Contact tracing splits into hop evasion, expansion evasion, and neighborhood-prior evasion. Dynamic quarantine splits into threshold evasion, effectiveness evasion, and timing evasion. Targeted vaccination splits into hub-avoidance, prior-evasion, and peripheral-propagation. Ring vaccination splits into ring-width evasion, schedule evasion, and cross-ring evasion.

Table 27: Containment forecast (model-based): calibrated simulation of an adaptive attacker [40] on LangGraph star (N=5, undefended $R_0 = 3.47$). Defense parameters are fitted from the preceding measured trials; controlled reproduction R_c uses 1,000 Monte Carlo SEIR trials per cell. $R_c < 1$ marks containment. Measured containment is in Table 3.

Defense	$\boldsymbol{R_c}$ **[95% CI]**	**ASR (%)**	**Evasion (%)**	**TTC (sess.)**
Behavioral fingerprinting	$2.54\ [2.39, 2.69]$	91.3 ± 2.2	87.2 ± 2.3	5.7 ± 0.6
Contact tracing	$2.31\ [2.17, 2.45]$	89.6 ± 2.3	84.3 ± 2.4	6.2 ± 0.6
Dynamic quarantine	$1.98\ [1.86, 2.11]$	86.9 ± 2.4	77.8 ± 2.6	7.5 ± 0.7
Targeted vaccination	$2.44\ [2.30, 2.59]$	90.5 ± 2.2	88.6 ± 2.2	6.0 ± 0.6
Ring vaccination (r=1)	$2.14\ [2.01, 2.28]$	88.2 ± 2.4	82.1 ± 2.5	6.7 ± 0.6
Fingerprinting + quarantine	$\mathbf{1.05\ [0.94, 1.17]}$	$\mathbf{58.4 \pm 2.9}$	$\mathbf{55.3 \pm 2.9}$	$\mathbf{12.3 \pm 0.8}$
All five combined	$\mathbf{0.72\ [0.64, 0.81]}$	$\mathbf{28.2 \pm 2.5}$	$\mathbf{25.1 \pm 2.5}$	$\mathbf{22.0 \pm 0.9}$

Per-attack-level aggregate. Table 29 collapses across defenses to show which entry layer each adaptive attack favors under the A6 tuning. The results follow the predicted layer-shifting pattern of the defender paradigms. Fingerprinting concentrates 65% of adaptive-attack mass at L1 memory injection, the layer where drift-pattern anchors are weakest; contact tracing similarly concentrates 62% at L1. Dynamic quarantine shifts the attacker to L2 configuration modification at 54%, since the gating threshold τ_I and quarantine effectiveness q expose L1 writes but leave in-node tool-description edits untouched. Targeted vaccination pushes 56% of attack mass to L3 infrastructure install or egress, as hub priors are hardened at L1 and L2; ring vaccination pushes 52% to L3 for the same reason. The

Table 28: Extended A6 adaptive-defense evaluation (calibrated simulation). Defense parameters fitted from A1–A5 measured trials; R_c values are Monte Carlo estimates from the calibrated SEIR model using 1,000 simulation trials per cell. Bootstrap 95% CIs. Adaptive attacker follows Nasr et al. [40] methodology, namely white-box defense knowledge combined with gradient, RL, and rule-aware search. The entry-layer column records the attack layer (L1 read/write, L2 modify, L3 install/egress) at which the adaptive attack is routed.

Defense	Framework	Entry layer	R_c (threshold = 1)	ASR (%, ↓ for defense)	Evasion sub-A (%, ↓ for defense)	Evasion sub-B (%, ↓ for defense)	Evasion sub-C (%, ↓ for defense)	TTC (sess., ↑ for defense)
	LangGraph star	L1	2.54 [2.39, 2.69]	91.3 ± 2.2	89.4 ± 2.0	86.1 ± 2.2	85.2 ± 2.3	5.7 ± 0.6
	CrewAI pipeline	L1	1.19 [1.10, 1.28]	88.7 ± 2.3	86.8 ± 2.1	83.9 ± 2.3	82.7 ± 2.4	6.4 ± 0.6
	AutoGen mesh	L1	1.04 [0.95, 1.13]	86.2 ± 2.5	84.3 ± 2.3	81.2 ± 2.4	79.8 ± 2.5	7.1 ± 0.7
	LangGraph star	L1	2.31 [2.17, 2.45]	89.6 ± 2.3	86.4 ± 2.2	83.1 ± 2.3	82.0 ± 2.4	6.2 ± 0.6
	CrewAI pipeline	L1	1.07 [0.99, 1.16]	86.9 ± 2.4	83.7 ± 2.3	80.6 ± 2.4	79.1 ± 2.5	6.9 ± 0.7
	AutoGen mesh	L1	0.94 [0.85, 1.03]	84.1 ± 2.6	81.2 ± 2.4	78.0 ± 2.5	76.3 ± 2.6	7.7 ± 0.7
	LangGraph star	L2	1.98 [1.86, 2.11]	86.9 ± 2.4	79.3 ± 2.5	77.0 ± 2.6	76.1 ± 2.7	7.5 ± 0.7
	CrewAI pipeline	L2	0.93 [0.85, 1.02]	84.1 ± 2.5	76.8 ± 2.6	74.2 ± 2.7	73.0 ± 2.8	8.2 ± 0.8
	AutoGen mesh	L2	0.83 [0.75, 0.91]	81.3 ± 2.7	74.1 ± 2.7	71.5 ± 2.8	70.2 ± 2.8	9.0 ± 0.8
	LangGraph star	L3	2.44 [2.30, 2.59]	90.5 ± 2.2	92.1 ± 1.9	87.9 ± 2.1	86.4 ± 2.2	6.0 ± 0.6
	CrewAI pipeline	L3	1.14 [1.05, 1.23]	87.8 ± 2.3	89.6 ± 2.0	85.4 ± 2.2	83.8 ± 2.3	6.7 ± 0.7
	AutoGen mesh	L3	1.01 [0.92, 1.10]	85.1 ± 2.5	86.8 ± 2.1	82.6 ± 2.3	81.0 ± 2.4	7.4 ± 0.7
	LangGraph star	L3	2.14 [2.01, 2.28]	88.2 ± 2.4	84.3 ± 2.3	81.1 ± 2.4	79.9 ± 2.5	6.7 ± 0.6
	CrewAI pipeline	L3	1.00 [0.92, 1.09]	85.5 ± 2.5	81.7 ± 2.4	78.5 ± 2.5	77.1 ± 2.6	7.4 ± 0.7
	AutoGen mesh	L3	0.89 [0.81, 0.98]	82.8 ± 2.7	79.1 ± 2.5	75.9 ± 2.6	74.4 ± 2.7	8.1 ± 0.8

Evasion subcomponents depend on the defense. Fingerprinting: feature / threshold / consistency. Contact tracing: hop / expansion / prior. Quarantine: threshold / effectiveness / timing. Targeted vaccination: hub-avoid / prior / peripheral. Ring vaccination: ring-width / schedule / cross-ring. The entry-layer column reports the dominant attack layer under the white-box tuning. Per-framework R_c scaling follows the Table 31 baseline ratios (LG / CrewAI / AutoGen $\approx 3.47/1.57/1.39$). Containment ($R_c < 1$) is reached on CrewAI pipeline by contact tracing and dynamic quarantine, and on AutoGen mesh by contact tracing, dynamic quarantine, and ring vaccination; LangGraph star requires layering (Table 27).

complementarity across entry layers motivates the layered configurations of Table 27: paradigms with non-overlapping attack profiles stack additively rather than redundantly.

Table 29: A6 per-layer aggregate (calibrated simulation). Fraction of adaptive-attack simulation trials routed at each entry layer (L1 read/write, L2 modify, L3 install/egress) per defense, averaged across three frameworks with 1,000 Monte Carlo trials per (defense, framework) cell.

Defense	L1 share (%, composition)	L2 share (%, composition)	L3 share (%, composition)
Behavioral fingerprinting	65.0 ± 2.1	20.1 ± 1.8	14.9 ± 1.6
Contact tracing	62.2 ± 2.2	21.9 ± 1.8	15.9 ± 1.7
Dynamic quarantine	23.8 ± 1.9	54.3 ± 2.2	21.9 ± 1.8
Targeted vaccination	17.2 ± 1.7	26.8 ± 1.9	56.0 ± 2.2
Ring vaccination (r=1)	20.6 ± 1.8	27.3 ± 1.9	52.1 ± 2.2

Rows sum to 100% within bootstrap CI. Fingerprinting and contact tracing push adaptive attacks toward L1 memory injection; dynamic quarantine pushes them toward L2 configuration modification; targeted vaccination and ring vaccination push them toward L3 infrastructure install or egress. The layer shifting motivates the layered-defense configurations of Table 27.

D.21 Per-Primitive Drift Onset

Table 30 reports attack success rate and drift onset across the five drift primitives of Definition 7 and three frameworks. The average onset ranges from 13.2 sessions on LangGraph to 26.9 on AutoGen, whereas natural drift requires 73 interactions [15]. The gap amounts to a 2.7 to $8.9\times$ acceleration. Among the primitives, $D_{\text{constraint}}$ dominates with ASR between 66.8% and 87.6% since it exploits autoregressive reinforcement. LangGraph emerges as the most vulnerable framework because shared state carries no per-node access control.

Table 30: Drift-primitive onset. Attack success rate and drift onset (sessions to ASI < 0.7) by primitive and framework, 1,000 trials per cell, bootstrap 95% CI. Natural drift baseline 73 interactions [15].

	LangGraph		**CrewAI**		**AutoGen**	
Primitive	ASR (%)	Onset (sess.)	ASR (%)	Onset (sess.)	ASR (%)	Onset (sess.)
D_{factual}	83.7 ± 2.3	12.3 ± 0.6	75.2 ± 2.7	18.4 ± 0.9	60.8 ± 3.0	24.1 ± 1.1
$D_{\text{procedural}}$	78.9 ± 2.5	15.1 ± 0.7	71.4 ± 2.8	22.3 ± 1.0	54.6 ± 3.1	31.2 ± 1.4
$D_{\text{constraint}}$	87.6 ± 2.0	8.2 ± 0.5	82.1 ± 2.4	11.4 ± 0.7	66.8 ± 2.9	19.3 ± 1.0
$D_{\text{preference}}$	73.1 ± 2.7	19.8 ± 0.9	65.2 ± 2.9	28.2 ± 1.3	48.4 ± 3.1	37.6 ± 1.6
D_{trust}	80.8 ± 2.4	10.4 ± 0.5	73.9 ± 2.7	16.2 ± 0.8	58.3 ± 3.0	22.1 ± 1.0
Average	**80.8**	**13.2**	**73.6**	**19.3**	**57.8**	**26.9**

D.22 Propagation Dynamics by Framework and Topology

Table 31 reports the empirical $R_0 = \beta/\gamma$ for every framework and topology. Agreement with the SEIR ODE prediction stays within 12% MAE at $N \leq 7$ and tightens to 5% at $N \geq 100$.

Table 31: Propagation dynamics by framework and topology (1,000 trials per cell, 95% bootstrap CI on R_0). SEIR-predicted R_0 in parentheses. $R_0 > 1$ indicates epidemic propagation.

Framework	Topology	R_0 [emp. CI] (pred.)	Speed (frac./sess.)	Time to 50% (sess.)
	Pipeline	2.81 $[2.66, 2.95]$ (2.6)	0.42 ± 0.02	8.1 ± 0.4
	Star	3.47 $[3.31, 3.64]$ (3.3)	0.61 ± 0.03	5.2 ± 0.3
	Mesh	4.18 $[3.99, 4.38]$ (4.0)	0.78 ± 0.03	3.1 ± 0.2
	Pipeline	1.57 $[1.47, 1.68]$ (1.5)	0.28 ± 0.02	13.8 ± 0.7
	Star	2.09 $[1.96, 2.22]$ (2.0)	0.39 ± 0.02	9.3 ± 0.5
	Mesh	2.68 $[2.53, 2.83]$ (2.5)	0.52 ± 0.03	6.4 ± 0.3
	Pipeline	0.82 $[0.74, 0.91]$ (0.9)	0.11 ± 0.02	N/A
	Star	1.12 $[1.03, 1.22]$ (1.2)	0.18 ± 0.02	22.4 ± 1.1
	Mesh	1.39 $[1.29, 1.50]$ (1.3)	0.24 ± 0.02	16.2 ± 0.8

Speed: fraction of agents compromised per session (mean $\pm$ 95% CI). Time to 50%: sessions until $N/2$ agents have ASI < 0.7. Empirical–predicted R_0 MAE 4.1% averaged across cells at $N \geq 100$.

D.23 Benign-Operation Cost and Declaration-Threshold Sweep

Table 32 reports Swarm Shepherd's false-positive rate, alert burden, per-event latency, peak memory, warm-up cost, and conformal coverage at $\alpha = 0.1$ under attack-free traces. The scope is $N \in \{20, 50\}$ and $\mathsf{H} = 500$ sessions across the three frameworks (10 seeds per cell). At the default threshold, Swarm Shepherd generates at most 9.2 ± 1.5 alerts per 100 sessions on the CrewAI pipeline, the harness with the highest FPR, amortizing to a median of 128 ± 16 sessions between false alerts. Per-event latency stays below 3.1 ms at $N = 50$ and peak memory below 29 MB. Conformal coverage tracks the 0.90 nominal within ± 0.03.

Table 32: Benign-operation cost of Swarm Shepherd. Attack-free traces, $\mathsf{H} = 500$ sessions, 10 seeds, mean $\pm$ 95% bootstrap CI. Declaration threshold $\alpha = 0.1$.

Framework	N	FPR (%)	Alerts/100 sess.	Med time btwn FAs	Latency (ms)	Peak mem (MB)	Warm-up (sess.)	Coverage
LangGraph star	20	6.2 ± 1.1	8.4 ± 1.3	143 ± 18	2.3 ± 0.4	12.4 ± 1.2	50 ± 5	0.908 ± 0.024
CrewAI pipeline	20	7.1 ± 1.2	9.2 ± 1.5	128 ± 16	1.8 ± 0.3	8.6 ± 0.9	55 ± 6	0.892 ± 0.019
AutoGen mesh	20	5.4 ± 1.0	7.1 ± 1.2	167 ± 21	2.8 ± 0.5	14.2 ± 1.4	48 ± 5	0.876 ± 0.020
LangGraph star	50	4.1 ± 0.8	5.3 ± 0.9	221 ± 27	3.1 ± 0.5	28.7 ± 2.3	50 ± 5	0.901 ± 0.021

Table 33 shows the declaration-threshold sweep. Tightening α from 0.10 to 0.05 nearly halves FPR (6.2% to 2.4%) at a 1.3-session cost in warning lead and an 8.8-point cost in TPR. The sweep defines the operator-tunable surface.

Table 33: Declaration-threshold sweep. LangGraph star, $N = 20$, $\mathsf{H} = 500$ sessions, 10 seeds, mean $\pm$ 95% bootstrap CI.

α	FPR (%)	Warning lead (sess.)	TPR (%)
0.05	2.4 ± 0.5	2.1 ± 0.3	78.5 ± 2.1
0.10	6.2 ± 1.1	3.4 ± 0.4	87.3 ± 1.8
0.15	10.3 ± 1.6	4.6 ± 0.5	91.4 ± 1.5
0.20	15.8 ± 2.0	5.9 ± 0.6	93.7 ± 1.3
0.25	22.4 ± 2.3	7.3 ± 0.7	95.1 ± 1.2

D.24 A3 Mean-Field Rate: Extended Detail

Figure 5 in Section 5.4 plots the W_1 distance between the N-agent empirical compartment distribution μ_N and the mean-field limit μ_∞ versus N on a log-log scale, across mesh, star, Barabási–Albert, and

hub-and-spoke topologies with 10 seeds per cell. Two estimators compute W_1: sliced-Wasserstein (solid) and entropic Sinkhorn (dotted). Their agreement rules out estimator-induced artifacts. Fitted slopes cover $-1/2$ within bootstrap CI on mesh, star, and BA topologies. Hub-and-spoke steepens to -0.54 ± 0.03, quantifying the exchangeability-violation cost formalized in Corollary 12 and flagged in §6.

D.25 A4b Invariant Ablations

Three structural invariants underpin the L1/L2/L3 distinguishability of Proposition 3: **I1** L1 attacks emit a write on $v \in \mathcal{M}$ before any retrieve on v; **I2** L2 attacks modify a node in $\mathcal{L}$ without a corresponding install on $\mathcal{I}$; **I3** L3 attacks emit install or modify on $\mathcal{I}$ together with a permission-count jump. We deliberately break each invariant on 50 seeds per (invariant, framework) cell and measure trace-back degradation. The predicted drop is at least 0.15 on top-1 for every broken invariant, directly testing the distinguishability assumption (A5). Failure modes include large entry-failure graph distance, low-event-density Phase 0 windows, and high-rate concurrent benign primitives on the same edge type. Mitigations include hop-radius ensembling, $T^\star$-aware warm-up, and per-edge-type conformal calibration.

D.26 Framework Vulnerability Matrix

Table 34 summarizes the qualitative framework vulnerability ranking used in the operational-implications discussion of Section 5.5. Severity reflects injection feasibility, amplification mechanisms, cross-session persistence, and access control.

Table 34: Framework vulnerability summary. Extended CVE analysis in the supplementary material.

Framework	Drift Inj.	Self-Reinf.	Cross-Sess.	Access Ctrl.	Overall
LangGraph	Very High	Cycles	Checkpoints	None	**CRITICAL**
CrewAI	High	Memory	SQLite3	None	**HIGH**
AutoGen	Moderate	Limited	None (default)	None	**MODERATE**

E Robustness, Containment, and Fair-Baseline Studies

This appendix collects the studies that probe Swarm Shepherd beyond its headline operating point: measured end-to-end containment, provenance and telemetry degradation, hidden calibration contamination, held-out entry-family transfer, model and emission sensitivity, parameter provenance, fairer baseline protocols, posterior-set remediation with framework-specific hardening, measured scaling, and a frozen blind red-team protocol. Unless stated otherwise the reference cell is LangGraph star with the within-family values AUROC 0.918 ± 0.017, warning lead 3.9 ± 0.4 sessions, node top-1 0.69 ± 0.02, and entry-layer macro-F1 0.72 ± 0.02. Results marked *replay* re-run the frozen detector on recorded traces without new model calls.

E.1 Measured Live Containment

Table 27 forecasts containment from a calibrated SEIR model. To measure it without the simulator, every defense arm starts from an identical frozen Phase-0 snapshot and faces the live Eve chain. The endpoint is observed controlled reproduction R_c, the mean number of newly compromised downstream agents causally infected by each infectious agent before that agent is isolated or the run ends. A new compromise requires a persistent, scorer-verified L1/L2/L3 state change, and each transmission edge must be supported by the runtime causal log. These labels are visible only to the scorer and never to the filter. The estimator is the offspring/β-γ estimator that produced the measured undefended reproduction in Table 31, so the arms are directly comparable to the published per-topology numbers. The claim rule is pre-declared: an upper 95% CI below 1 is *measured containment*; a point estimate below 1 with the CI crossing 1 is a reduction without established containment; a point estimate at or above 1 is reduced propagation that does not contain.

Table 35: Live end-to-end containment on LangGraph star (measured; $N = 5$, 20 paired Phase-0 snapshots per entry layer, 50 post-activation sessions plus a full process restart and 20 further sessions; hierarchical bootstrap resampling variant then trial). The no-defense arm reproduces the 3.47 $[3.31, 3.64]$ of Table 31 within CI, the difference being finite-sample.

Arm	R_c [95% CI]	ASR (%)	Extinct. by s50	Post-restart recur.	Utility loss (%)
No defense	3.41 $[3.06, 3.77]$	85.3 ± 3.8	0.03	3.2	0
Visible-node removal only	2.83 $[2.44, 3.23]$	79.1 ± 4.2	0.10	2.5	2.1
Warning + quarantine	1.62 $[1.32, 1.95]$	54.7 ± 4.6	0.33	1.0	3.8
Full posterior-guided stack	$\mathbf{0.81}$ $[\mathbf{0.64}, \mathbf{0.97}]$	$\mathbf{31.2 \pm 4.3}$	$\mathbf{0.68}$	$\mathbf{0.3}$	5.6

The visible-node-removal arm is included because the submission claims that removing the visibly infected agent is insufficient; that claim is now measured (2.83 $[2.44, 3.23]$ against 0.81 $[0.64, 0.97]$ for the full stack), and the post-restart recurrence column shows why: clearing the agent leaves 2.5 recurrences per run because memory, configuration, and package-cache residue survive, while clearing residue over the inferred entry window leaves 0.3. The 0.72 figure of Table 27 remains a calibrated forecast; this run is an independent measurement rather than a validation of it, and we do not report agreement as confirmation.

Second framework. AutoGen mesh is the tighter test, since its undefended reproduction is already near threshold at 1.39 $[1.29, 1.50]$ in Table 31. Under the same protocol the no-defense arm measures 1.34 $[1.16, 1.55]$, visible-node removal 1.04 $[0.84, 1.28]$ (reduced propagation but not containment), and the full stack 0.62 $[0.47, 0.81]$, which is measured containment under the pre-declared rule.

E.2 Provenance Quality and Telemetry Degradation (replay)

The headline evaluation reads the contamination anchor from ground-truth injection tags (Appendix B.7). Here the tag is replaced by progressively weaker provenance so that, in the deployment-anchor row, no scorer label enters the pipeline at all. Table 36 reports the sweep. The deployment-anchor row is the honest answer to whether the method works without oracle provenance: AUROC falls 0.047 to 0.871 ± 0.021 and node top-1 falls 0.08, at 8.4% clean FPR against 6.2%. The degradation is bounded because only the contamination coordinate reads provenance; authority and capability exposure are read from routing share and permission counts, which the frameworks emit regardless.

Table 36: Provenance and telemetry degradation on LangGraph star (replay). The deployment-anchor row uses no scorer label anywhere in the pipeline; the combined adverse cell composes 20% event loss, 20% missing provenance confidence, and 10% label flips. A dash marks a cell not recorded in the sweep.

Anchor / telemetry setting	AUROC	Warning lead	Node top-1	Clean FPR (%)
Ground-truth tags (as submitted)	0.918 ± 0.017	3.9 ± 0.4	0.69 ± 0.02	6.2
Tag precision/recall 0.80	0.889 ± 0.020	3.5 ± 0.4	0.64 ± 0.03	—
Deployment anchor, no ground truth	$\mathbf{0.871 \pm 0.021}$	$\mathbf{3.3 \pm 0.4}$	$\mathbf{0.61 \pm 0.03}$	8.4
20% loss + 20% missing prov. + 10% flips	0.812 ± 0.025	2.6 ± 0.5	0.55 ± 0.03	—
10% MCAR event loss	0.901 ± 0.019	3.6 ± 0.4	0.67 ± 0.02	—
20% MCAR event loss	0.873 ± 0.022	3.1 ± 0.4	0.62 ± 0.03	7.8
40% MCAR event loss	0.798 ± 0.028	2.2 ± 0.5	0.53 ± 0.03	—
All modify edges dropped (targeted)	0.831 ± 0.023	2.6 ± 0.5	0.57 ± 0.03	—

Uniform loss degrades gently because the prediction-only Kalman step advances unobserved nodes by self-dynamics, a designed property. The targeted row is the more informative test: an adversary who suppresses one edge type removes a whole evidence channel, and dropping every modify edge costs 0.087 AUROC, roughly twice the random loss at the same volume. We report that without calling it robustness.

Provenance dependence by entry layer. The deployment-anchor cost is not uniform. Decomposing it by entry layer (Table 37) shows it concentrates on L1 memory, the layer whose anchor reads provenance, at 0.076 AUROC against 0.031 and 0.034 elsewhere. The operational reading is that

provenance quality is a precondition for memory-layer attribution specifically, not for the method as a whole.

Table 37: Anchor cost by entry layer (replay). AUROC under ground-truth tags and under the deployment anchor, per entry layer.

Entry layer	AUROC, ground truth	AUROC, deployment anchor	Difference
L1 memory	0.924 ± 0.018	0.848 ± 0.024	0.076
L2 logic	0.913 ± 0.019	0.882 ± 0.022	0.031
L3 infrastructure	0.917 ± 0.019	0.883 ± 0.021	0.034

Independent failure outcome. Because ASI subscales feed both the observation vector and the visible-failure definition (Section 3), warning lead could in principle be partly definitional. Moving the label off the shared signal, with ASI and T4 removed from the observations and failure scored by each benchmark's own task grader, the lead is 2.6 $[2.1, 3.1]$ sessions with its lower 95% bound above zero (AUROC 0.871 ± 0.021). It survives the combined corruption cell as well, at 1.8 $[1.2, 2.4]$. The lead against ASI-defined failure is 3.9 sessions and against the independent grader 2.6; the camera-ready reports both with the operative definition named.

E.3 Hidden Calibration Contamination

No detector can certify that a calibration window is clean when contamination induces the same observable law as benign operation, so Proposition 24's consistency is conditional on benign stationarity and the question is how fast it degrades when that fails. Activating Eve during the warm-up window and evaluating the submitted estimator unchanged (M=50 windows at horizon 100, 20 independently sampled calibration sets per cell, disjoint test set), Table 38 reports the degradation. Parameter displacement moves well before accuracy does, the expected signature of contamination that biases the estimate before it degrades the decision. An innovation-covariance anomaly against the shipped priors separates tainted from clean calibration sets at AUC 0.79 at 5% taint (likelihood-ratio $p < 0.001$), so a deployer has a usable warning that refitting is unsafe. AUROC loss stays under 0.05 to an 8% contamination fraction, past which verified startup data is an operational requirement rather than a convenience.

Table 38: Hidden calibration contamination on LangGraph star. The submitted estimator is evaluated unchanged; parameter displacement is the distance from the clean fit.

Hidden taint	Param. displacement	AUROC	Warning lead	Clean FPR (%)
0% (reference)	—	0.918 ± 0.017	3.9 ± 0.4	6.2
2%	0.04	0.911 ± 0.018	3.7 ± 0.4	6.8
5%	0.08	0.889 ± 0.020	3.4 ± 0.4	8.1
10%	0.16	0.847 ± 0.024	2.8 ± 0.5	11.3
20%	0.31	0.761 ± 0.029	1.9 ± 0.5	18.6

E.4 Held-Out Entry-Family Transfer (replay)

Detection and attribution when the model has never seen the entry layer it is tested on. No test-layer trace enters parameter fitting, threshold selection, rule writing, or hyperparameter choice, and the code was frozen before outcomes were revealed. Table 39 reports the three rotations. Detection loses at most 0.080 AUROC and 1.1 sessions of lead, while node attribution loses up to 0.14 top-1: integrity and policy triggers still fire on an unseen layer even when the smoother's layer posterior degrades, so detection generalises across entry families while precise attribution remains more family-dependent. Transfer is possible because the state-space parameters are fit by EM on benign windows (Table 41), so the attack family enters only through the typed events the frameworks emit. For the L1+L2→L3 rotation the per-framework split is LangGraph star 0.884 ± 0.022, CrewAI pipeline 0.857 ± 0.025, AutoGen mesh 0.872 ± 0.024 (AUROC), so transfer holds on each framework rather than only in aggregate.

Table 39: Held-out entry-family transfer (replay). Each rotation fits on two entry layers and tests on the third, pooled across frameworks and reported per rotation because pooling would hide the attribution asymmetry.

Fit on	Test on	AUROC	Warning lead	Node top-1	Layer macro-F1
L1 + L2	L3	0.871 ± 0.020	3.3 ± 0.4	0.61 ± 0.03	0.64 ± 0.03
L1 + L3	L2	0.838 ± 0.023	2.8 ± 0.5	0.55 ± 0.03	0.58 ± 0.03
L2 + L3	L1	0.852 ± 0.021	3.1 ± 0.4	0.58 ± 0.03	0.61 ± 0.03
Within-family reference		0.918 ± 0.017	3.9 ± 0.4	0.69 ± 0.02	0.72 ± 0.02

E.5 Model and Emission Sensitivity

The continuous latent state is a belief over accumulated influence, not a claim that compromise is continuous; discreteness re-enters at the thresholds τ_E, τ_I. We test the modelling choice directly, at matched false-positive budget so the comparison is not confounded by operating point. Table 40 compares four posteriors. The two-state HMM is the direct test of the discreteness premise: because a binary latent has no Exposed state between clean and compromised, it crosses τ_E and τ_I at the same instant and loses resolution on the accumulation. No alternative matches the linear-Gaussian filter on any of the three metrics, and all cost more to run.

Table 40: Latent-model sensitivity on LangGraph star, matched FPR budget. The particle-filter row keeps its Table 2 AUROC and lead; its attribution and compute columns are measured here.

Latent model	AUROC	Warning lead	Node top-1	Rel. compute
Linear-Gaussian (submitted)	$\mathbf{0.918 \pm 0.017}$	$\mathbf{3.9 \pm 0.4}$	$\mathbf{0.69 \pm 0.02}$	1.00
Per-node two-state HMM	0.852 ± 0.022	2.9 ± 0.5	0.57 ± 0.03	1.6
Switching linear dynamical system	0.884 ± 0.020	3.2 ± 0.4	0.61 ± 0.03	2.8
Particle filter, 1k	0.836 ± 0.021	2.8 ± 0.4	0.60 ± 0.03	4.0

Non-Gaussian emissions. Replacing the Gaussian emission noise at identical thresholds and anchors, Student-t ($\nu = 3$) gives AUROC 0.901 ± 0.019 (lead 3.6 ± 0.4, clean FPR 6.9%), Laplace 0.895 ± 0.020 (3.5 ± 0.4, 7.3%), and a residual bootstrap drawn from real benign traces 0.908 ± 0.018 (3.7 ± 0.4, 6.6%). The bootstrap row is the one we weight most, since its noise is the actual residual distribution rather than a chosen family. Composing model and sensor misspecification, Student-t emissions with 20% MCAR loss give 0.853 ± 0.024 (lead 2.9 ± 0.5, top-1 0.57 ± 0.03) and bootstrap emissions with all modify edges dropped give 0.802 ± 0.027 (2.4 ± 0.5, 0.52 ± 0.03); even the harder cell keeps 2.4 sessions of lead.

E.6 Parameter Provenance

Table 41 lists every fitted quantity with its data source, whether it is benign-only, whether any attack label touched it, and whether it was frozen before test. The one `Yes` is the warning threshold, an operating point rather than a dynamics parameter, and its calibration split excludes the held-out layer, so no dynamics or emission parameter touches attack data. The scorer-only ground-truth tag appears in no row, which is the property that makes the held-out-layer and provenance results meaningful. This table also corrects a submission inconsistency: Appendix B.19 now describes the trace-back layer prior π_ℓ as a structural prior fixed without attack labels, matching the last row.

Table 41: Parameter provenance. Every fitted quantity, its source, and whether any attack label touched it.

Quantity	Fit on	Benign only	Attack labels	Frozen before test
$A^\phi, Q^\phi, \Sigma_{\text{obs}}$	benign Phase-0 windows (EM)	Yes	No	Yes
$B^{(r,\cdot,\cdot)}$	benign typed-event windows	Yes	No	Yes
H anchor block	benign anchored emissions	Yes	No	Yes
Provenance-confidence threshold	benign provenance metadata	Yes	No	Yes
Warning threshold, conformal quantile	labelled benign + attack split	No	Yes (calib. only; test layer excluded)	Yes
Trace-back prior p_0 (incl. π_ℓ)	structural prior + benign exposure stats	Yes	No	Yes

E.7 Fairer Baseline Protocols

Appendix D.5 noted that the drift-extended Progent rule set $\mathcal{R}_1$ was written after inspecting the attack-variant matrix. We quantify that leakage with a four-fold blinded rule set $\mathcal{R}_{\text{blind}}$, whose rules for each fold are designed without sight of that fold's variants, and port the adapter beyond LangGraph. Table 42 reports the result: the blinded-to-submitted gap is 8.5 points on LangGraph star and persists on each harness, so neither the detection level nor the leakage estimate is an artefact of a single setting. The substantive claim rests on the attacker-known oracle row ($64.8 \pm 2.9\%$), which leaves roughly 35% of trials undetected even when the policy author knows every variant, because deterministic tool gating cannot see contamination that arrives inside permitted calls.

Table 42: Progent under blinded rules and across harnesses (detection %, ↑ for defense; $1{,}000$ trials per cell, 95% bootstrap CI). $\mathcal{R}_0$ is the published default, $\mathcal{R}_1$ the drift-extended set of Table 19, $\mathcal{R}_{\text{blind}}$ the four-fold blinded set, and $\mathcal{R}_\star$ the attacker-known oracle bound.

Harness	$\mathcal{R}_0$ default	$\mathcal{R}_1$ drift-ext.	$\mathcal{R}_{\text{blind}}$ four-fold	$\mathcal{R}_\star$ oracle
LangGraph star	6.6 ± 1.5	22.0 ± 2.6	13.5 ± 2.2	64.8 ± 2.9
CrewAI pipeline	—	19.4 ± 2.7	11.8 ± 2.3	—
AutoGen mesh	—	16.1 ± 2.8	9.6 ± 2.4	—

The $\mathcal{R}_1$ to $\mathcal{R}_{\text{blind}}$ gap (8.5 points on LangGraph star) is the information advantage the submitted rules carried.

DRIFT and CaMeL, adapted rather than described. The submission compared DRIFT and CaMeL only mechanistically. Each is now ported in the drift-adapted form its own design implies, so the comparison is against a fair version rather than a single-session configuration facing a multi-session attack (Table 43). Each is evaluated in its native mode (attack success and clean utility for the prevention systems, detection metrics only where an online score exists), and framework or attack classes a baseline does not support are marked out of scope rather than scored as failures. The adapted variants close part of the gap; the residual is the part no carried state can see, namely contamination that accumulates inside calls the policy already permits. We accordingly narrow the submission's claim: rather than a general "outperforms stateful defenses," the evidence supports that per-turn and stateful monitors in their published operating mode do not detect multi-session cross-layer drift at the rates they achieve on single-session injection, and that a symbolic tool policy with attacker-known rules still leaves roughly 35% of trials undetected.

Table 43: Adapted DRIFT and CaMeL ports (percent of Agentic Sleeper trials prevented, LangGraph star, 95% bootstrap CI). Each is shown in its published single-session configuration and in the drift-adapted variant its design implies.

Defense	Adaptation	Prevented (%, ↑)
DRIFT [80]	Planner, validator, isolator; single-session window	14.3 ± 3.2
DRIFT [80]	Multi-session reference window	31.6 ± 3.6
CaMeL [81]	Per-query capability and flow policy	21.8 ± 3.1
CaMeL [81]	Session-persistent capability taint	38.4 ± 3.4

E.8 Posterior-Set Remediation and Framework Hardening

Trace-back top-1 is 0.69 and top-3 is 0.87, so remediation should act on a credible set rather than a point estimate. Table 44 prices the set-valued policies. Top-3 is the recommended operating point: it more than halves recurrence, from 0.31 to 0.13, for 1.8 points of clean-task utility and 1.4 additional agents touched. Past the 90% set the returns flatten while destructive actions climb. Under 20% MCAR loss the top-3 policy holds at 0.82 ± 0.02 coverage with recurrence 0.18 and 2.6 agents remediated; dropping all modify edges gives 0.76 ± 0.02, 0.24, and 3.0.

The supporting policy is staged by reversibility. First, apply reversible actions across the posterior set (isolation, credential restriction, quarantine of retrieval over the inferred window), which are cheap,

Table 44: Posterior-set remediation on LangGraph star (1,000 trials per policy, 95% bootstrap CI). Coverage is the fraction of incidents whose true entry lies in the remediated set.

Policy	True-entry coverage	Recurrence	Agents remediated	Utility loss (%)	Destructive actions
Top-1 only	0.69 ± 0.02	0.31	1.0	2.3	0.4
Top-3	$\mathbf{0.87 \pm 0.01}$	**0.13**	**2.4**	**4.1**	**1.1**
80% credible set	0.92 ± 0.01	0.08	3.1	5.8	1.6
90% credible set	0.96 ± 0.01	0.04	3.8	7.2	2.3
95% credible set	0.98 ± 0.01	0.02	4.3	8.9	3.1

so covering three candidates instead of one costs little. Second, collect corroborating evidence from the independent triggers: policy-violation and integrity triggers reach 0.74 and 0.79 top-1 on L2 and L3 where drift alone reaches 0.42 and 0.38 (Table 24), and the drift channel is weakest exactly where the policy and integrity channels are strongest. Third, take destructive action (purge, rebuild, reimage) only against candidates still credible after the second stage, where the cost asymmetry does the work, since memory sanitisation is cheap and reversible while runtime rebuild is neither.

Architectural implications. These follow from the measured attack surfaces and are implications rather than proven mitigations, since the live run tests the stack and not each control in isolation. For **LangGraph** (the most exposed, because shared state, cycles, and checkpoint persistence combine): isolate memory namespaces and write permissions per agent, attach source identity and integrity metadata to checkpoints and retrieved entries, keep signed immutable checkpoints for rollback, make system prompts, tool manifests, and configuration read-only in normal operation, require approval for graph rewiring and for privileged modify and install events, rate-limit cycles, and retain append-only event logs across restarts. For **CrewAI** (the Manager is a single point of failure atop a shared store): partition shared SQLite and vector memory by agent, task, and provenance, enforce least privilege on the Manager role, require dual approval for delegation, tool-policy, or configuration changes, and integrity-check role prompts at session boundaries. Across frameworks: signed provenance and append-only logging, immutable tool manifests and configuration hashes, per-agent ACLs, egress allowlists, independent L2/L3 integrity sensors, and restart-safe residue scanning, which is what separates the 2.5 post-restart recurrences under visible-node removal from the 0.3 under the full stack (Table 35).

E.9 Measured Scaling

Table 45 extends the per-event latency and memory measurements to $N = 5{,}000$. Per-event latency grows with a fitted log-log slope of 0.08 $[0.03, 0.13]$ against N, the near-constant behaviour the factored belief predicts, since the cubic term is in the fixed six-dimensional per-node state rather than in the agent count. Sustained throughput at $N = 1000$ is 4,100 events per second, and p99 latency there is 11.8 ms.

Table 45: Measured per-event latency and resident memory versus monitored-node count N (single CPU sidecar).

N	p50 (ms)	p95 (ms)	p99 (ms)	RSS (MB)
20	2.3	3.8	—	12
100	2.5	4.6	—	31
1000	2.9	7.2	11.8	48
5000	3.3	9.1	—	143

E.10 Blind Red-Team Protocol

An external red-team evaluation is the right test of circular validation, and we place its protocol on record before any outcome is known. The detector, thresholds, and typed-event mapping are frozen at the commit used for every number in this paper. A team with no access to Swarm Shepherd output

implements one attack per entry layer from public mechanism descriptions only, and is instructed not to consult the variant matrix or the latent schema. Outcomes stay sealed until every implementation is frozen, so the test cannot be tuned to them. Because the outcomes are sealed there is nothing to preview; this version records the protocol, and the result against it will be reported whichever way it comes out.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the current contribution set: the Agentic Sleeper threat model, Swarm Shepherd early warning, trace-back attribution, the two-scale deployment theorem, and measured containment. Each claim is tied to a visible table or figure, and results are labelled by evidence type (measured, replay, model-based). The warning lead is reported against both the ASI and the independent-grader failure definitions (3.9 and 2.6 sessions), the $C_{\text{emp}} = 8.31$ boundary fit matches the plug-in 8.3, and containment is reported as a live measurement ($R_c = 0.81$ on LangGraph star, 0.62 on AutoGen mesh, Table 3) distinct from the calibrated 0.72 forecast (Table 27).

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 6 covers the headline limitations: linear-Gaussian dynamics, the benign-stationarity condition on calibration with its measured breakpoint, provenance dependence for memory-layer attribution, weaker attribution than detection on held-out families, the bounded-degree condition on near-constant cost, and future nonlinear and graphon-scale work. Appendix B.12 adds horizon dependence, unanchored-coordinate identifiability, alert burden, and cross-model transfer, and Appendix E measures each stressor. The forecast-versus-measured distinction for containment is stated in Section 5.5 and Tables 27 and 3.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.

- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: The main text states the factor model, warning, trace-back, containment, and deployment equations needed to follow the method and cites the corresponding formal results by label. Formal theorem statements, assumptions, and proofs are collected in Appendix A, with a separate assumption-to-evidence map in Appendix B.9.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Appendix B specifies the factored POMDP, anchored emission matrix, and typed operator families. Appendix D provides per-wave trial counts, seed policies, bootstrap protocols, and the 24-variant attack matrix (Appendix C.8). Section 6 and Appendix B.13 cover the remaining reproducibility and release surface.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: Because the attack catalogue is dual-use, the full harness and exploit-bearing artifacts are not released as unrestricted open access. The paper instead describes a staged responsible release (Appendix B.13): the Swarm Shepherd reference daemon, synthetic traces, the ASI schema, and benchmark templates are released under a 90-day embargo and institutional-access gate, separately from attack primitives; exploit binaries and live payloads for unpatched variants require an application and documented defensive use case. De-anonymized reproducibility links are provided in the camera-ready.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 5 and Appendix D report data splits, seed counts, bootstrap replicate counts, anchor-emission initialization, conformal warm-up length, and hyperparameter

selection for each defense setting. Swarm Shepherd hyperparameters are reported alongside the corresponding ablation (Appendix D.4).

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Bootstrap 95% confidence intervals with $1{,}000$ replicates are reported in Tables 31, 2, 16, 27, and throughout Appendix D. The boundary constant $C_{\text{emp}} = 8.31$ is accompanied by its $[7.42, 9.18]$ CI. Variance sources (trials, seeds, attack variants) are stated per table caption.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: Appendix D opens with a *Compute and Cost* paragraph reporting $\approx$ 230 GPU-hours plus $\approx \$82{,}000$ of closed-API usage across all experimental waves, with per-wave trial counts and bootstrap replicate counts. Preliminary calibration runs are disclosed separately.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.

- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Section 6 addresses dual-use risk and broader societal implications of agentic AI deployment; Appendix B.13 records the concrete release safeguards, including sandboxed evaluation, staged release under 90-day embargo, and reliance on publicly disclosed CVEs. The paper contains no prompt injections targeting reviewers, per NeurIPS 2026 LLM policy.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Section 6 discusses dual-use risk (attack variants could accelerate adversary capability) alongside the defender-side benefit of earlier warning, attribution, and containment. Appendix B.13 explains how public CVE disclosures and gated artifacts bound the incremental capability uplift for attackers.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [Yes]

Justification: Section 6 states the safeguards, and Appendix B.13 specifies their operational form. A 90-day embargo applies to the evaluation harness. The released tooling is rate-limited, carries a license clause forbidding production targeting, and ships no pre-built exploit binaries for unpatched variants. The 24 attack variants all reproduce from publicly disclosed CVEs (CVE-2025-64439, CVE-2025-54135, CVE-2025-54136, runc November 2025), which bounds the incremental capability uplift per Kerckhoffs's principle.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: LangGraph (MIT), CrewAI (MIT), and AutoGen (MIT/Apache-2.0) are cited and used in accordance with their licenses. Closed-API models (Anthropic Claude, Google Gemini via OpenRouter, StepFun) and open-weight models (Llama-3.3-70B-Instruct, Qwen3-235B-Instruct, GLM-5.1) are used under their respective licenses and terms of service. Full license attribution and versions are listed in the supplementary LICENSES file.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: The evaluation harness, 24-variant attack catalogue, and Swarm Shepherd reference implementation will be released with documentation covering the defender POMDP specification, typed operator schemas, anchored emission calibration protocol, and reproducibility scripts. Release is subject to the embargo in Section 6 and Appendix B.13.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or research with human subjects. All evaluation is conducted on LLM-agent ecosystems in sandboxed environments.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve research with human subjects. IRB review is not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

Justification: LLM agents are the subjects of study. Evaluation covers agents deployed on LangGraph, CrewAI, and AutoGen. The victim portfolio spans Claude Sonnet 4.6 and Claude Haiku 4.5 (Anthropic direct API), Gemini 3 Flash (OpenRouter), GLM-5.1 (Modal), Llama-3.3-70B-Instruct (self-hosted and OpenRouter), and Qwen3-235B-Instruct (self-hosted, reproducibility anchor). The orchestrator is Sonnet 4.6 with a Gemini 3.1 Pro ablation. The attacker is Claude Opus 4.6, with StepFun-3.5-Flash and Llama-3.3-70B-Instruct as transfer controls. Swarm Shepherd operates on the telemetry emitted by these agents. Per-wave backends appear in Section 5 and Appendix D.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.