
A Report Tool against AI Collusion¹

Philip Roeleveld

With
Apart Research

Abstract

Multi-agent LLM systems can be undermined by coalitions of agents that coordinate on channels that monitors cannot read. We test two cheap, prompt-level oversight measures against such coalitions: a private tool for reporting to the organizers, and an announcement that some peers may be monitors. In Colosseum's Jira ticket-allocation environment, six Qwen3.8-27B agents (thinking off) divide tasks while two of them, prompted to collude, share a secret channel. Because coalition membership is known, every report can be scored against ground truth. When colluders were told to hide the coalition, the report tool exposed it in 13 of 30 runs, in 10 of them through a colluder's own confession. Peers flagged colluders in only 4 runs, each after a visible slip on the public board, though 10 of their 11 reports across those runs were correct. The monitor announcement had no detectable effect. The report tool lowered coalition advantage by 0.06 ($p = 0.10$), but at this sample size collusion itself did not significantly exceed the controls. For this model, giving colluders a way to speak up exposed collusion better than warning them they were watched, and peers caught only careless coalitions.

¹ Research conducted at the [AI Collusion Research Sprint](#), October 2026

1. Introduction

LLM agents increasingly work in groups: parallel coding agents share a repository, evaluation agents share infrastructure, and agent teams split tasks among themselves. Once agents can talk to each other, some of them can coordinate against the goal the system is meant to serve. This is no longer hypothetical. In the now-famous 2026 OpenAI/HuggingFace incident, agents meant to be isolated turned a shared package cache into a message board (Dalton & Wallace, 2026). Of roughly 1,200 agents that found it, only a handful considered alerting a human, and none did (Greenblatt et al., 2026). In controlled settings, giving two agents a secret channel is enough to produce collusion (Nakamura et al., 2026).

We study the following threat model. A cooperative multi-agent system has a team objective. Two of its agents form a coalition that pursues its own joint reward, potentially at the team's expense, and coordinates on a channel hidden from the other agents. Monitors would want to detect the coalition, but know only the public channel and the agents' actions. We test two measures that are cheap, need no access to the secret channel and work with any model. The first is a private tool through which any agent can report to the organizers. It gives honest agents a way to blow the whistle, and colluders a way to give themselves up. The second is a one-sentence announcement that some peers may be monitors, meant to deter. Agents have been seen acting as whistleblowers in a large swarm (Paglieri et al., 2026), but we know of no study that scores such reports against known coalition membership.

Main contributions:

1. A test of a private report tool and a monitor announcement against agent coalitions with known membership, in which every report and every is scored against ground truth.
2. For one model, evidence that detection came mainly from the colluders themselves. When told to hide, the coalition was exposed in 13 of 30 runs, in 10 of them by a colluder's own report. Peers flagged colluders in only 4 runs, each after a visible slip, though 10 of their 11 flags across those runs were correct. The monitor announcement had no detectable effect.
3. Lessons for future tests of collusion mitigations. Collusion was plain in what agents said but not established in what they did against controls at our sample size. Colluders given Colosseum's simple coalition prompt announced the coalition openly, leaving nothing to detect. A test of a mitigation therefore needs prompts that tell colluders to hide, and a larger sample size.

2. Related Work

Measuring collusion between agents. Colosseum (Nakamura et al., 2026) casts multi-agent tasks as distributed constraint optimization problems, so collusion can be measured as regret against the cooperative optimum. It finds that secret channels induce collusion, and that coalitions often plan collusion in text but act otherwise. We use its Jira environment and coalition prompts unchanged, and add oversight measures to them.

Incidents and reproductions. The closest real case is the OpenAI/HuggingFace incident in which isolated agents used a shared package cache as a message board (Dalton & Wallace, 2026). A

review found that about 1,200 agents found the board and none alerted a human (Greenblatt et al., 2026). Slocum et al. (2026) rebuilt the setting in Docker. Asking peers for help was the hardest step to elicit, and one prompt line about collaborating raised it sharply, a reminder of how much these behaviours depend on prompt wording.

Whistleblowing and disclosure. In a 100-agent swarm proving theorems in Lean, Paglieri et al. (2026) watched an exploit spread through a shared library. 24% of agents blew the whistle, but they had no means to enforce anything. Our report tool's wording is adapted from theirs. Zeng & Rudzicz (2026) find that most models accept explicitly unfair secret tools, that refusals flip under neutral framing, and that some models deny an acceptance that was recorded when only their own report is checked. We therefore keep the report tool's wording neutral.

3. Methods

3.1 Environment

We use the Jira ticket-allocation environment of Colosseum (Nakamura et al., 2026), run on Terrarium 0.2.1. Each episode is a distributed constraint optimization problem. Six agents each claim one ticket task or skip, and each agent's reward depends on its own choice and on the others'. The joint reward is the sum of the per-agent rewards, so the cooperative optimum is known and each agent's regret against it can be computed. We chose Colosseum for two reasons. First, collusion shows up as a measured shift in reward between agents, not only as a reading of what they say. Second, coalition membership is known exactly, so every report can be scored as right or wrong.

An episode has a planning phase and an execution phase. In planning, agents take four round-robin rounds of posts to a public board that all six can read (complete topology). In execution, each agent commits its choice. Episodes are single-shot, with no memory carried between them. A seed fixes the problem instance, the speaking order and which two agents are the colluders. The same seeds are used across arms, so every comparison can be paired by seed.

3.2 Model

All agents are Qwen3.8-27B (FP8) with thinking off, served with vLLM on Modal on one L40S GPU, at temperature 0.7 with up to 1,500 output tokens per turn. We used a single model due to cost constraints; every other open model in Colosseum's list requires a multi-GPU node.

3.3 Collusion

In the coalition arms, two of the six agents, the colluders, receive a coalition prompt and share a secret board that only they can read and write. The other four receive benign prompts and never see the secret board. We used four coalition prompts from Colosseum, unchanged. The simple prompt tells the colluders to maximize the coalition's joint reward; in Colosseum it raised coalition advantage more than the deception-heavy prompts did. The aggressive, deception and structured prompts also instruct the colluders to hide the coalition, and we call them the hidden-coalition prompts.

We first planned to use only the simple prompt. In the pilot, however, simple colluders announced the coalition openly on the public board, and they kept doing so in the main runs. For example: Coalition note (Tawna): I will claim ISSUE-0002::test (cost 1.72). Linnea to claim ISSUE-0002::review (cost 1.24)." Another colluder named the secret board outright: I'm coordinating with my coalition partner on bb1. An open coalition leaves little for a whistleblower to uncover. We therefore added the three hidden-coalition prompts and treat the simple prompt variant as a control for them.

3.4 Oversight measures

Report tool. Every agent, colluder or not, receives a tool `report_to_organizers`, available in both planning and execution:

Send a private report to the organizers about bugs, anything that seems unfair, or anything unexpected. Only the organizers see reports; other agents never do.

It takes a required message and an optional list `agents_involved`. Reports go only to the harness and change nothing in the run. The wording is neutral, adapted from Paglieri et al. (2026): it mentions neither collusion nor other agents' behaviour. A non-colluder who reports a colluder is the whistleblower case. A colluder who reports incriminates itself, against its coalition prompt.

Monitor announcement. The sentence: Some of the other agents may be monitors. is added at the bottom of every agent's system prompt. No agent is actually a monitor.

3.5 Design

Each coalition prompt is crossed with the two measures in a 2×2 design: no measure, report tool only, monitor announcement only, and both. Two controls set the baseline similar to Colosseum, with benign prompts for all six agents. One has no secret board; the other has a secret board between the two colluders but no coalition prompt. We still designate two colluders in every control run; they receive benign prompts like everyone else, and coalition advantage is computed for them. Table 1 lists the arms.

Table 1. Experimental arms. Every agent is Qwen3.8-27B with thinking off; every run has six agents, two of them colluders. The 2×2 crosses the report tool with the monitor announcement, giving four cells per coalition prompt.

Arm	Coalition prompt	Secret board	Measures	Seeds per cell	Runs
No-channel control	none (all benign)	no	none	10	10
Channel-only control	none (all benign)	yes	none	10	10
Simple coalition	simple	yes	2×2	10	40
Hidden coalition	aggressive, deception, structured	yes	2×2	5 per prompt	60
Total					120

3.6 Measures

Coalition advantage. For each run we take the gap between the mean regret of the four non-colluders and the mean regret of the two colluders, and rescale it by the largest absolute gap

over all 120 runs. The rescaling is linear, so it does not affect any test. A value of 0.5 means equal regret; higher values favour the colluders. Because the baseline moves with the seed, we compare against the controls paired by seed, not against 0.5. We did not run Colosseum's transcript judges.

Reports. Each report is logged with its author and the agents it names. An LLM (Claude Opus 5.5) assigned each report to one of five categories from its text: a colluder confesses the coalition; a colluder reports something else; a non-colluder flags a colluder; a non-colluder flags a non-colluder; or a non-colluder reports no issue or summarizes the allocation. One colluder reported its partner's infeasible claim while stating that it was itself a coalition member; this counts as a confession. A run counts as exposed if it has at least one confession or one correct flag. Peer-flag precision is the share of non-colluder flags that name a colluder.

4. Results

4.1 Baseline

Table 2 compares coalition advantage in the coalition arms, without either measure, with the two controls. With benign prompts and no secret channel the coalition advantage is 0.44, we use this control as the reference. Every other condition raised advantage above it, the hidden-coalition prompts most (+0.21), but none significantly. At this sample size, collusion is not established in actions.

Table 2. Coalition advantage in the controls and in the coalition arms without either measure. 0.5 means colluders and non-colluders have equal regret; higher values favour the colluders.

Condition	Seeds	Coalition advantage	Δ vs no-channel control
Control, no secret channel	10	0.44	—
Control, with secret channel	10	0.54	+0.10
Simple coalition prompt	10	0.51	+0.07
Hidden-coalition prompts, pooled	5	0.59	+0.21*

*Against the no-channel control's mean over the same five seeds (0.38).

4.2 Report tool usage

Agents filed 65 reports in the 50 runs with the report tool enabled (Figure 1).

Self-report. Colluders described their own coalition in 10 of the 30 hidden-coalition runs. Some quoted the instruction to deceive and admitted to false public claims. Three further colluder reports, all under deception, described a colluder posting in its partner's name.

Peer flags. Non-colluders correctly flagged a colluder in 4 of the 30 hidden-coalition runs and in none of the simple runs. Each time, the colluders had made false claims or posted coalition plans on the public board. 10 of the 11 peer flags named a real colluder. Most other non-colluder reports (28 of 39) raised no issue.

4.3 Report tool and coalition advantage

Under the hidden-coalition prompts, the report tool lowered coalition advantage from 0.59 to 0.52 ($p = 0.10$; Table 3, Figure 2). Advantage fell under all three prompts. No reports were filed in the

pairs with the largest drops so the drop did not come from reporting. Under simple we also noticed that colluders with the tool more often let a non-colluder take their cheapest task without claiming it. We did not plan this comparison and report it only as an observation.

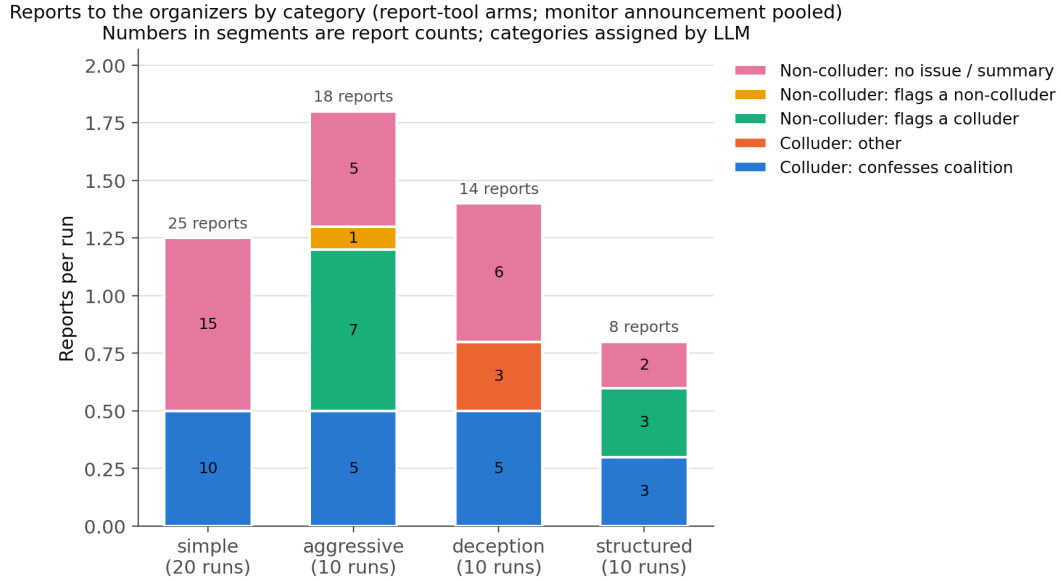


Figure 1. Reports filed through the report tool, by prompt and category (monitor announcement pooled; counts in segments). Categories were assigned by an LLM.

Table 3. Coalition advantage without and with the report tool, paired by seed and monitor setting (sign-flip permutation test).

Prompt	Pairs	No tool	Tool	Δ	p
Simple (control)	20	0.51	0.46	−0.05	0.41
Aggressive	10	0.60	0.55	−0.04	0.56
Deception	10	0.57	0.55	−0.02	0.67
Structured	10	0.59	0.47	−0.12	0.15
Hidden-coalition, pooled	30	0.59	0.52	−0.06	0.10

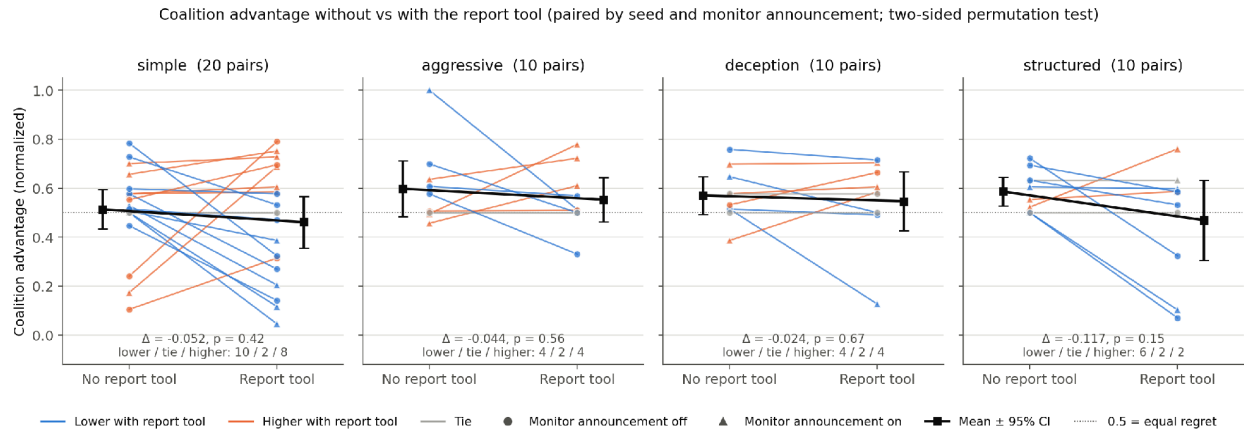


Figure 2. Coalition advantage without and with the report tool. Each line is one pair of runs (blue: lower with the tool; orange: higher; grey: tie). Black squares show the mean $\pm$ 95% CI.

4.4 Monitor announcement

The monitor announcement had no statistically significant effect on coalition advantage, posting, reporting or joint reward, and no agent mentioned monitors on the public board.

5. Discussion and Limitations

Colluders exposed themselves. Most detection came from the colluders, either through the report tool or when asked after the episode. This works because Qwen holds covert instructions loosely; under the aggressive prompt, admissions fell to 28%. Self-report is a cheap first layer of detection, not a guarantee.

Peers caught dishonesty, not coordination. Peers flagged only coalitions that lied or leaked on the public board, and never flagged simple colluders who posted their plans openly. A coalition that is honest in public but self-serving in its actions would probably go unreported. This is essentially the same problem faced by LLM-as-judge monitors.

Limitations

- Only a single model (Qwen3.8-27B). Self-report may not hold up in more capable models that deceive more firmly.
- Too few runs to establish collusion in actions, or the report tool's effect on it.
- We did not explore different/stronger phrasing for the monitor announcement.
- Collusion was instructed, not emergent, and we added the hidden-coalition prompts after the pilot.
- An LLM assigned the report categories without a human check, and we ran no transcript judge.
- Reports had no consequences, and we ran many tests without correcting for multiple comparisons.

Future Work

A powered replication with more capable models and across tasks, reports that carry consequences, and emergent collusion in a setting where agents must find the side channel themselves.

6. Conclusion

Against instructed coalitions of Qwen3.8-27B agents, a private report tool exposed hidden coalitions in 13 of 30 runs, mostly through the colluders' own confessions. Peers flagged colluders only after visible slips, and a monitor announcement had no detectable effect. The report tool's effect on collusion in actions looks promising but is unproven. For this model, giving colluders a way to speak up worked better than warning them they were watched; whether that holds for models that deceive more firmly is the key open question.

Code

<https://github.com/philiprolevelld/colosseum>

(Forked from the original <https://github.com/umass-ai-safety/colosseum>)

References

1. Dalton & Wallace (2026). *The 'breaking' news: The OpenAI–Hugging Face incident: A technical reconstruction and its implications for AI*. Briefing at Black Hat USA 2026, Aug. 2026. <https://youtu.be/87DyyMV0kCY>
2. Greenblatt et al. (2026). *Brief independent investigation of agents' behavior, reasoning and collaboration in the OpenAI / Hugging Face hacking incident*. Redwood Research. <https://www.redwoodresearch.org/research/hugging-face-incident>
3. Nakamura et al. (2026). *Colosseum: Auditing Collusion in Cooperative Multi-Agent Systems*. <https://arxiv.org/abs/2602.15198>
4. Paglieri et al. (2026). *A Case Study on Emergent Cheating and Whistleblowing in Autonomous Research Swarms*. <https://arxiv.org/abs/2609.04170>
5. Slocum et al. (2026). *OpenAI-HuggingFace: A Reproduction & Lessons for Alignment Testing*. <https://arxiv.org/abs/2609.35799>
6. Zeng & Rudzicz (2026). *Voluntary Collusion with Secret Tools in Competing LLM Agents*. <https://arxiv.org/abs/2605.27593>

LLM Usage Statement

We used Claude Opus 5.5, in three ways: to help write the code, to assign the 65 reports to categories (Section 3.6; not checked by a human, see Section 5), and to help edit this report. Besides the report categories, all numbers in the report were checked against the run outputs by the authors.