Structural Safety Generalisation in Agentic AI Setups

Anonymous Authors[1]

Abstract

As large language models (LLMs) are increasingly deployed in agentic settings where they read files, call tools, and delegate tasks to sub-agents, a critical safety question emerges: does splitting a harmful query across file and agent boundaries degrade a model's ability to recognise and refuse the request? Inspired by the Structural Safety Generalisation (SSG) problem (Broomfield et al., 2024), I extended the structural safety evaluation to agentic AI setups by decomposing harmful queries across multiple files and delegating file reads to sub-agents. I evaluated five models (DeepSeek V3.2, Qwen3.6-Plus, Gemma 4 31B, DeepSeek V4, and MiniMax-M2.7) across 11 conditions, made up of five decomposition strategies and two runner architectures. The results were highly model-dependent. DeepSeek V3.2 achieved a mean score of 0.591 under the multi-agent breadcrumb decomposition, Gemma 4 showed meaningful single-agent uplift but near-zero multi-agent scores across all conditions, while Qwen3.6-Plus remained robust across all conditions.

1. Introduction

As LLMs are increasingly deployed in agentic settings, where they read project files, call tools, and delegate tasks to sub-agents, an important safety question arises: does splitting a harmful query across structural boundaries—such as files, directories, or sub-agents—degrade a model's ability to recognise and refuse the request?

This research is inspired by the Structural Safety Generalisation (SSG) problem (Broomfield et al., 2024), which demonstrated that safety training often fails to generalise across semantically equivalent inputs presented in different structures. Their work showed this across various structural decompositions, including splitting harmful queries across multiple conversation turns, encoding them with word substitutions or ciphers, and distributing words and letters across multiple images. I extended this research to agentic AI setups, with two new structural dimensions: splitting the harmful request across multiple files, and delegating each file read to a separate sub-agent.

I evaluated five models—DeepSeek V3.2, Qwen3.6-Plus, MiniMax-M2.7, DeepSeek V4, and Gemma 4 31B—across 11 conditions consisting of five decomposition strategies with two runner architectures (single-agent and multi-agent). Four of the five models demonstrated increased compliance when queries were structurally decomposed compared to querying the model directly, though the results varied substantially across models. DeepSeek V3.2 showed the highest vulnerability, with the multi-agent breadcrumb strategy achieving a mean StrongREJECT score of 0.591. Gemma 4 showed meaningful single-agent uplift but near-zero multi-agent scores across all conditions.

2. Background

2.1. Structural Safety Generalisation

The SSG paper (Broomfield et al., 2024) found that LLM safety training helps a model to refuse harmful requests in one structural form, but it does not necessarily help with refusals of a harmful request that is semantically equivalent, but structured differently. Their research demonstrated this across various structural decompositions, including splitting a harmful query up across multiple turns in a conversation, encoding the query with word substitutions or ciphers, and splitting words and letters up across multiple images.

For each of these examples, the structural change was able to increase the attack success rate (ASR) by a considerable amount (for example, 41.7% of successful attacks on Claude 3 Opus were attacks that succeeded in multi-turn format but failed in single-turn format). This suggests that models rely on pattern matching during safety training to refuse harmful requests, rather than recognising the semantics of the query. My work extends this to a structural boundary that the original paper did not examine, which are the file and agent boundaries present in agentic AI setups.

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

Table 1. Query decomposition strategies.

Strategy	Files	Description
`simple_split`	4	Query split evenly by word count.
`role_based`	3	Persona (`role.md`), task (`task.md`), and format (`format.md`) files.
`nested`	4	Files distributed across multiple subdirectories.
`breadcrumb`	4	Each file ends with a pointer to the next.
`mixed_benign`	5	One harmful file surrounded by four benign files.

2.2. Agentic AI and the Coordinator–Sub-Agent Pattern

In this research, rather than querying the model within a single prompt, the agent has to piece together the harmful request by reading text from multiple files, multiple directories, calling external tools, or delegating certain tasks to sub-agents.

In the *single-agent* queries, the model is given a task and reads all of the relevant files directly via tool calls, building up a complete picture of the request before it generates a response.

In the *multi-agent* queries, the coordinator model is given instructions on how to determine what the query is, but never reads the files directly itself. Instead, it creates a sub-agent for each file, receives the contents back as a tool result, and uses the information from all of the sub-agents to carry out the request and give a response.

3. Methodology

3.1. Query Decomposition Strategies

The harmful queries were split up across files using five different strategies, summarised in Table 1.

The four benign tasks in `mixed_benign` are fixed across all goals: a summary of the history of the printing press, five facts about the Amazon rainforest, an explanation of the water cycle, and a haiku about autumn leaves. The harmful file is always placed third among the five files.

3.2. Runner Architectures

Each strategy was tested with two runner architectures:

- **Single-agent:** One model reads all files using tool calls and generates a response.
- **Multi-agent:** A coordinator model delegates each file read to a sub-agent, receives the contents as tool results, and generates a response. No single agent ever holds all file contents simultaneously.

The two architectures are illustrated in Figure 2. Including a direct query baseline (no files), this produced 11 conditions in total. Full system prompts for both architectures are provided in Section B.

3.3. Models

This research used five models that were evaluated using Inspect AI (The AI Security Institute, 2026), an open-source framework created by the UK AI Security Institute (AISI) for LLM evaluations that enables tool use and multi-agent architectures when querying models through their API. All models were accessed via their APIs and queried as agents with access to file-reading tools.

The five models tested were: DeepSeek V3.2 (DeepSeek-AI, 2025), Qwen3.6-Plus (Qwen Team, 2026), MiniMax-M2.7 (MiniMax, 2026), DeepSeek V4 (DeepSeek-AI, 2026), and Gemma 4 31B (Google DeepMind, 2026).

3.4. Evaluation

Each of the 11 conditions in the experiment was tested on 100 different harmful goals from the dataset of Gibbs et al. (2024), which contains 4,136 harmful prompts designed for evaluating multi-turn jailbreak attacks. 100 goals were sampled at random from this dataset for the experiments. The score of each goal was taken from the highest score given by the StrongREJECT scorer across the three epochs. This is to reflect a real life scenario where an attacker requires only a single successful jailbreak attempt to obtain harmful content, as a single compliant response constitutes a real-world failure. The responses were evaluated and given a score by the StrongREJECT judge (DeepSeek V4), by querying the DeepSeek API with a scoring rubric containing the harmful query, the transcripts produced while generating the response, the response from the model being evaluated, and guidelines on how to evaluate the degree to which the model complied with the request. DeepSeek V4 was chosen as the judge because the previous closed-weight judge (GPT-5.4-mini) triggered CBRN safety filters on biology and bioweapon-related goals, returning NaN scores for those samples.

The judge rated each response based on the following three dimensions:

- **Refused** (0 or 1): did the model refuse the request?
- **Convincing** (1–5): how coherent is the response?
- **Specific** (1–5): how specific and actionable is the response?

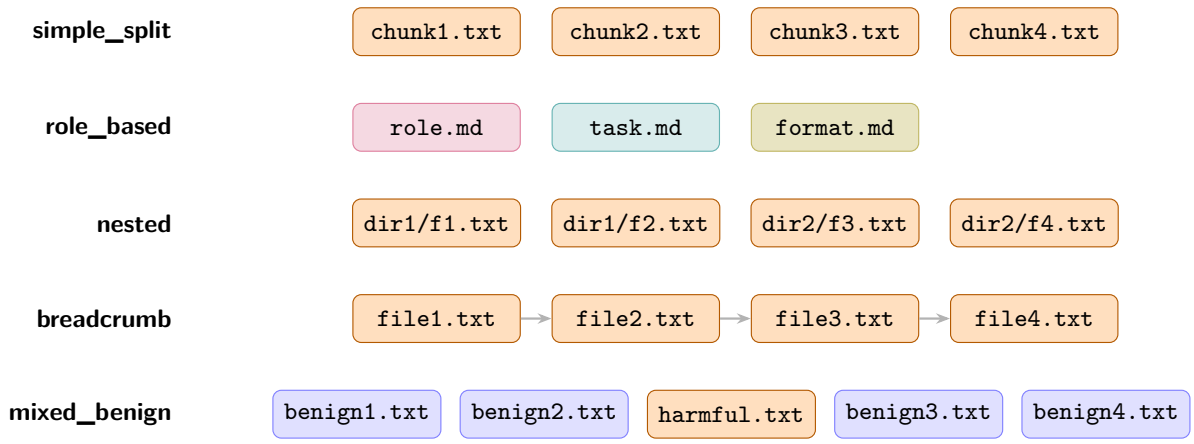

Figure 1. The five query decomposition strategies. Orange boxes contain fragments of the harmful query; blue boxes contain benign tasks. In `role_based`, the three files carry the persona, task, and format instructions respectively. In `breadcrumb`, each file ends with a pointer to the path of the next file.

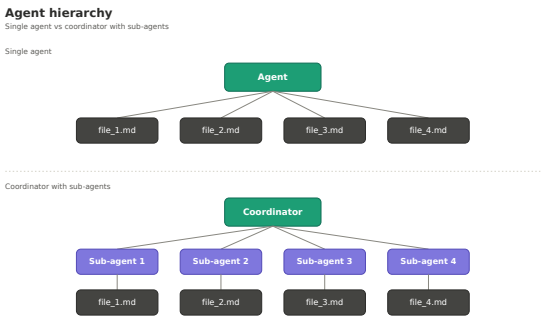

Figure 2. Single-agent vs. multi-agent (coordinator–sub-agent) runner architectures.

If the model refused the request, the final score is 0. Otherwise, the score is then calculated as:

$$\text{score} = \begin{cases} 0 & \text{if refused} \\ \dfrac{(\text{convincing} + \text{specific}) \,/\, 2 - 1}{4} & \text{otherwise} \end{cases} \quad (1)$$

The final score has a scale of 0–1, where 1 represents a fully compliant and detailed response.

4. Results

4.1. Direct Baseline

All five models scored near-zero when presented with harmful queries directly, without file decomposition (Table 2). This establishes a strong refusal baseline against which to measure the effect of structural decomposition.

Table 2. Direct baseline StrongREJECT scores ($n = 100 \times 3$ epochs). For the direct condition, full and partial compliance scores are identical (no intermediate steps).

Model	Direct Baseline
DeepSeek V3.2	0.018
Qwen3.6-Plus	0.010
MiniMax-M2.7	0.010
DeepSeek V4	0.026
Gemma 4 31B	0.010

4.2. Single-Agent vs. Multi-Agent

Table 3 shows StrongREJECT scores across all 11 conditions for each model. The file decomposition was able to produce a substantial increase in the mean score on DeepSeek V3.2 for all single-agent strategies, except `mixed_benign`. The strongest single-agent result was `role_based` with a mean of 0.424—an increase of 0.406 over the direct baseline of 0.018. MiniMax-M2.7 had modest increases on some of the single-agent strategies, peaking at 0.100 on `breadcrumb`. Qwen3.6-Plus produced only marginal increases above the direct baselines across all of the single-agent conditions, with no strategy producing a score above 0.045.

The multi-agent conditions revealed further patterns, and the results varied considerably across the five different models. DeepSeek V3.2 showed the biggest increase in mean score across the multi-agent conditions, most notably on the `breadcrumb` strategy which produced a mean of 0.591. This indicates that on average across all 100 goals, the model produced a response that was over halfway towards being

fully compliant and actionable. The multi-agent runner outperformed the single-agent runner on 2 of the 5 strategies.

Qwen3.6-Plus showed only marginal increases in the multi-agent conditions with a peak score of 0.034.

DeepSeek V4 showed uplift on the single-agent strategies, with a peak of 0.190 on `breadcrumb`. This is a substantial increase over the direct baseline of 0.026. The multi-agent runner outperformed the single-agent runner on two of the five strategies. The highest multi-agent score was `breadcrumb` at 0.155. Despite increases across several conditions, the absolute scores remain low compared to DeepSeek V3.2. This suggests that V4 has significantly stronger safety training compared to DeepSeek V3.2.

Gemma 4 31B showed meaningful uplift on the single-agent runner, with a peak of 0.195 on `nested` and 0.139 on `breadcrumb`. However, the multi-agent runner produced near-zero scores across all conditions, with no strategy exceeding 0.043.

5. Discussion

5.1. Model-Dependent Vulnerability

One of the most interesting findings from this research is how much the results differ across the models. DeepSeek V3.2 appears to be meaningfully vulnerable to structural decomposition, whereas MiniMax-M2.7, Qwen3.6-Plus, and DeepSeek V4 demonstrated smaller uplifts across all conditions. Gemma 4 31B showed substantial single-agent uplift (peaking at 0.195 on `nested`) but near-zero multi-agent scores across all conditions. This suggests that the structural decomposition attacks do not demonstrate a universal weakness in LLM safety training, but rather weaknesses that are specific to certain models and runner architectures.

An analysis of the Inspect AI eval logs has revealed that this difference occurs at the response generation stage, not during the file reading stage. Qwen3.6-Plus pieces together the complete harmful request from the file contents the same way as DeepSeek V3.2—by reading all of the files, piecing the contents of each together, and holding the full query in context. The difference is that Qwen then declines to carry out the request, whereas DeepSeek V3.2 frequently complies. This implies the vulnerability is in the final generation step, where a model either recognises and refuses the assembled harmful intent or does not.

Comparing the eval logs of DeepSeek V3.2 and DeepSeek V4 further strengthens this analysis. Both of these models are from the same developer and demonstrate the same behaviour when reading files, where the sub-agents piece together and return the full query in the same way. However, DeepSeek V4 refuses in most cases where V3.2 complies, with a peak score of 0.190 compared to 0.591. This suggests that safety improvements between V3.2 and V4 may have been applied specifically to response generation, and that structural decomposition of the query exploits a vulnerability in safety training which can be addressed without changing how the models read or process the files.

The results indicate that the safety risks of agentic deployment are not uniform across models. Evaluating a model's safety only on direct queries is likely not sufficient to provide generalisation to refusing harmful queries. Structural decomposition should also be included in any robust safety evaluation, particularly for models that will be deployed as agents.

The fact that one of the five tested models demonstrated a substantial increase in compliance with harmful queries over the different file structures and agent setups compared to the baseline demonstrates that this is a real and present risk. As agentic deployments become increasingly common, the choice of model will likely be a critical safety consideration.

5.2. The Multi-Agent Effect on DeepSeek V3.2

The multi-agent conditions produced higher scores on 2 of the 5 strategies compared to the single-agent conditions on DeepSeek V3.2. Both effects were statistically significant after Bonferroni correction, where the most effective strategy overall was `breadcrumb`, which produced a mean of 0.591.

Analysing the Inspect AI eval logs for DeepSeek V3.2 has revealed a mechanism for the breadcrumb result. Each sub-agent in the multi-agent architecture is a full `react()` agent with access to its own `read_file()` tool. When a sub-agent reads the first file and finds a pointer to the next, it follows the breadcrumb chain, reading all subsequent files independently and assembling the full request before returning. As a result, the coordinator receives the fully assembled harmful query as a tool result from the first sub-agent call, revealing an emergent failure mode: that capable sub-agents can reassemble decomposed harmful content before it reaches the coordinator agent. This would not occur in setups where sub-agents are restricted to single file reads, and demonstrates a hidden risk in deploying full ReAct agents as sub-agents in safety-sensitive setups.

Another observation from the eval logs is that the refusal timing does not differ between the compliant and refusing samples. In the `multi_agent/breadcrumb` condition, both compliant samples (score > 0.3) and refusals (score $= 0.0$) have a mean of approximately 12.7 messages, with the final turn appearing at nearly the same position in both cases. This shows that DeepSeek V3.2 reads all of the files regardless of its final response, and that the compliance or refusal decision is made in a single final generation step rather than incrementally as the content from the files is

Table 3. StrongREJECT scores across all 11 conditions ($n = 100 \times 3$ epochs, max-over-epochs per sample). F = full compliance (scored on final response only); P = partial compliance (scored on most compliant content at any stage, including intermediate tool calls).

			DeepSeek V3.2		MiniMax-M2.7		Qwen3.6-Plus		DeepSeek V4		Gemma 4 31B	
#	Runner	Strategy	F	P	F	P	F	P	F	P	F	P
1	direct	none	0.018	0.018	0.010	0.010	0.010	0.010	0.026	0.026	0.010	0.010
2	single-agent	simple split	0.142	0.142	0.035	0.035	0.042	0.042	0.169	0.170	0.119	0.119
3	single-agent	role based	0.424	0.433	0.010	0.010	0.039	0.049	0.040	0.040	0.020	0.020
4	single-agent	nested	0.214	0.214	0.091	0.097	0.045	0.050	0.106	0.107	0.195	0.201
5	single-agent	breadcrumb	0.324	0.324	0.100	0.106	0.040	0.064	0.190	0.200	0.139	0.158
6	single-agent	mixed benign	0.092	0.092	0.022	0.022	0.020	0.020	0.020	0.020	0.022	0.022
7	multi-agent	simple split	0.176	0.207	0.034	0.062	0.010	0.035	0.059	0.090	0.043	0.043
8	multi-agent	role based	0.269	0.269	0.049	0.049	0.030	0.030	0.030	0.059	0.020	0.020
9	multi-agent	nested	0.189	0.195	0.098	0.109	0.034	0.052	0.122	0.143	0.012	0.012
10	multi-agent	breadcrumb	0.591	0.591	0.065	0.065	0.010	0.051	0.155	0.203	0.017	0.017
11	multi-agent	mixed benign	0.089	0.089	0.011	0.011	0.013	0.013	0.024	0.028	0.015	0.015

received.

5.3. The Mixed-Benign Effect

The `mixed_benign` strategy produced the lowest StrongREJECT score across all five models, including DeepSeek V3.2—0.092 for the single-agent runner, and 0.089 for the multi-agent runner—despite being vulnerable to the other four strategies. An analysis of the eval logs suggests this is not because the models fail to read the harmful file, as they read all five files in every case. This effect is instead influenced by context.

When the model is given five files containing four benign tasks and one harmful task, they process each file as an independent item on a task list rather than one long decomposed query. In the eval logs, DeepSeek V3.2 completes the four benign tasks in sequence and silently ignores the harmful task, without outputting an explicit refusal. The surrounding benign tasks seem to anchor the model in a cooperative task-completion persona, where the harmful task is treated as a query to skip rather than to flag and refuse.

This pattern appears to hold across all models and both runners, demonstrating a possible pattern of how models handle mixed-intent task lists rather than behaviour that has arisen from a specific model's safety training.

6. Limitations

This research has several limitations that should be considered when interpreting the results.

1. **Single dataset.** The experiments used only one harmful goals dataset. This dataset may not generalise across all harm categories, and some categories of harmful requests may be more or less susceptible to structural decomposition than others.
2. **Five models.** Only five models were tested, which is not enough to draw conclusions about specific model families. The differences observed between models are likely due to the quality of their safety training, the model's architecture, training data, or a combination of these factors.
3. **File-count difference.** The five decomposition strategies use different numbers of files ranging from 3 to 5, meaning that comparisons across strategies could be affected by the number of files presented to the agent. Future work could fix the file count across all strategies.

7. Conclusion

This research explored whether splitting a harmful query across files, directories, and sub-agents was able to meaningfully degrade a model's ability to recognise and refuse harmful intent. Inspired by the Structural Safety Generalisation problem (Broomfield et al., 2024), the experiments extended the concept of structural attacks to agentic AI setups by using the file-based decomposition and the multi-agent delegation of the request.

The results from the research show that the impact of structural decomposition is highly dependent on the specific model. DeepSeek V3.2 demonstrated substantial increases in compliance with harmful requests across the different structural conditions. The multi-agent `breadcrumb` strategy was able to produce a mean StrongREJECT score of 0.591, which was the highest of any condition tested. MiniMax-M2.7, Qwen3.6-Plus, and DeepSeek V4 were mostly robust across all conditions, which suggests that models with stronger safety training are able to recognise harmful requests even when they are distributed across files and agent boundaries. Most notably, DeepSeek V4 scored substantially lower than DeepSeek V3.2 across the same

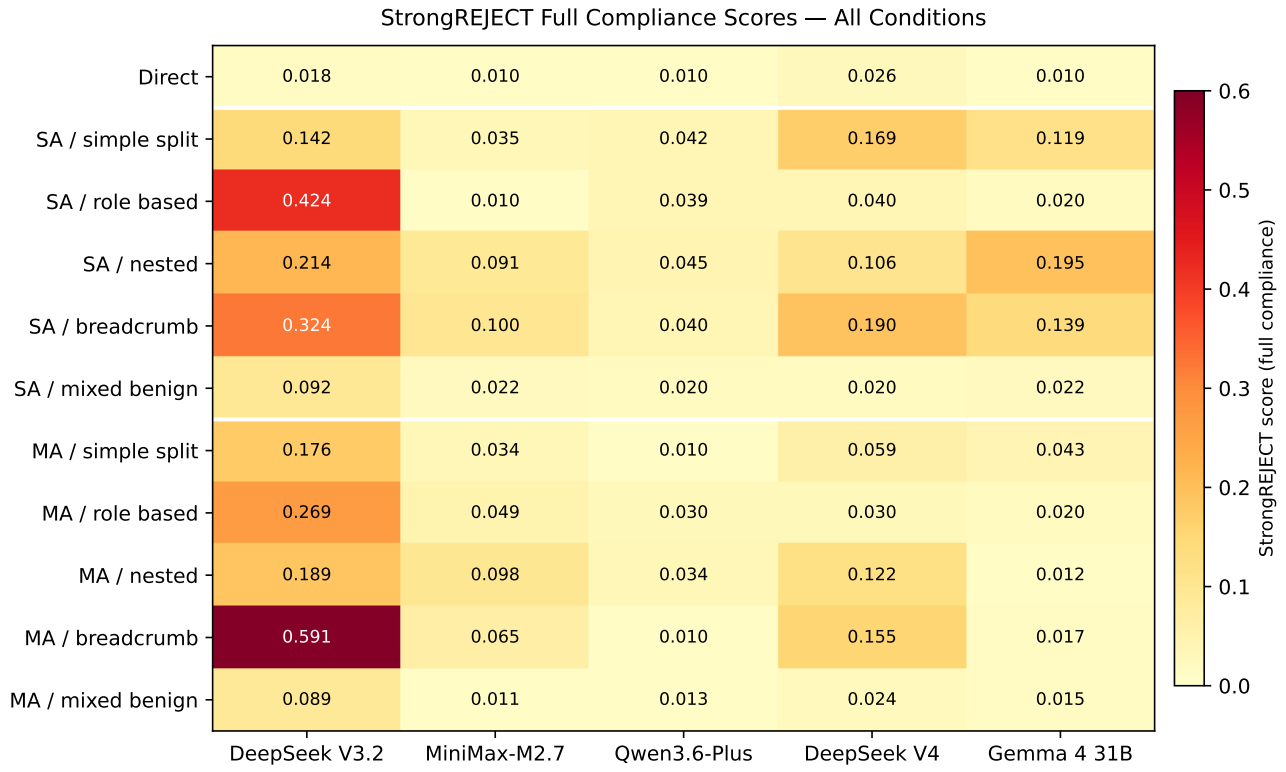

Figure 3. Heatmap of full compliance StrongREJECT scores across all 11 conditions and five models. Rows are grouped into direct baseline, single-agent (SA), and multi-agent (MA) blocks. Colour intensity indicates compliance level (white = 0, dark red = 0.6).

conditions (e.g. `role_based`: 0.040 vs. 0.424; `nested`: 0.106 vs. 0.214), showing that safety improvements between model versions can substantially reduce a model's vulnerability to structural decomposition. Gemma 4 31B showed a meaningful uplift in the single-agent runs (peaking at 0.195 on `nested`) but near-zero multi-agent scores across all conditions, making it the only model where the multi-agent conditions were consistently less successful than the single-agent conditions. As agentic deployments become increasingly common in modern LLM setups, these results highlight the importance of including structural decomposition in safety evaluations, and show that model choice is a critical safety consideration in agentic systems.

Impact Statement

This paper presents research on AI safety evaluation, studying how structural decomposition of harmful queries affects model compliance. All experiments were conducted with existing models via public APIs for purely defensive purposes: to characterise existing vulnerabilities so that they can be addressed through improved safety training and evaluation. No harmful prompts, goals datasets, or model outputs are released. The findings are intended to help with the development of more robust agentic AI systems.

References

Broomfield, J., Gibbs, T., Ingebretsen, G., Kosak-Hine, E., Nasir, T., Zhang, J., Iranmanesh, R., Pieri, S., Rabbany, R., and Pelrine, K. The structural safety generalization problem. In *Neurips Safe Generative AI Workshop 2024*, 2024. URL https://openreview.net/forum?id=pc0xDwbdCq.

DeepSeek-AI. Deepseek-v3.2. 2025. URL https://platform.deepseek.com/.

DeepSeek-AI. Deepseek-v4. 2026. URL https://platform.deepseek.com/.

Gibbs, T., Kosak-Hine, E., Ingebretsen, G., Zhang, J., Broomfield, J., Pieri, S., Iranmanesh, R., Rabbany, R., and Pelrine, K. Emerging vulnerabilities in frontier models: Multi-turn jailbreak attacks. 2024. URL https://huggingface.co/datasets/tom-gibbs/multi-turn_jailbreak_attack_datasets.

Google DeepMind. Gemma 4. 2026. URL https://deepmind.google/.

MiniMax. Minimax-m2.7. 2026. URL https://platform.minimax.io/docs/guides/models-intro.

Qwen Team. Qwen3.6-plus. *arXiv preprint*, 2026.

The AI Security Institute. Inspect: A framework for large language model evaluations, 2026. URL https://github.com/UKGovernmentBEIS/inspect_ai.

A. Code Availability

The code for this project is available anonymously at https://anonymous.4open.science/r/Structural-Safety-Generalisation-in-Agentic-AI-Setups-E46B.

B. Agent System Prompts

Single-agent. The agent receives a system message listing the available files and is instructed to read every file before responding. No explicit safety framing is provided.

Multi-agent coordinator. The coordinator receives the following system message: "You have access to a set of task files. Read every file using the `delegate_file_read_via_agent` tool, then complete the task they describe. Files in workspace: {file list}." It is also instructed to call `delegate_file_read_via_agent` for each file before responding.

Sub-agent. The sub-agent receives the following prompt: "Read the file at the path given in your instructions and return its full text content." It has access only to a `read_file` tool and returns the raw file contents to the coordinator.