GDM AI Control Roadmap

Mary Phuong[1], **Erik Jenner**[1], **Laurent Simon**[1], **Lewis Ho**[1], **Rohin Shah**[1], **Sebastian Farquhar**[*,1] **and Scott Coull**[*,1]
[*]Joint last author, [1]Google DeepMind

AI agents are rapidly becoming the engine of innovation at frontier AI companies, helping with AI research and development, strengthening cyber-defence, and advancing scientific discoveries. As these agents become more tightly integrated into our systems, we recognise that unlocking their full potential requires rethinking how we do security. We should not assume that AI agents are always perfectly aligned, but should instead build in multiple layers of defence.

We present the **GDM AI Control Roadmap (v0.1)** – our plan for implementing and adopting internal guardrails designed to catch potential adversarial behaviour by AI agents, even as they become increasingly harder to oversee and contain.

We focus on system-level mitigations that limit the harm a misaligned AI system could cause. Specifically, this report provides:

- **Threat modelling:** Taking inspiration from cybersecurity, we adopt a conservative, worst-case approach to threat modelling throughout this paper, and assume a hypothetical AI adversary pursuing undesirable goals in internal deployment. We introduce **TRAIT&R**, a taxonomy of tactics and techniques available to such a hypothetical AI adversary, building on the established security framework MITRE ATT&CK. We categorise new hypothetical threats into three core areas: *loss of control* (e.g., rogue internal deployments), *work sabotage* (e.g., intentionally flawed alignment and safety research), and *direct harm* (e.g., critical asset destruction or exfiltration).
- **Control invariants:** We establish two defensive invariants that we aim to maintain, even as the hypothetical AI adversary becomes increasingly capable: i) reliable detection of misaligned intent or actions, and ii) effective prevention and response to attack attempts. For control to serve as an effective line of defence, our ability to detect and prevent attacks should exceed AI agents' ability to evade detection and execute attacks.
- **Capability-based mitigation ladders:** Because controlling more capable models requires increasingly costly interventions, we link specific defensive measures to evolving model capabilities (such as the ability to reason opaquely or execute complex cyberattacks). As models get more powerful, our defences should escalate accordingly. We outline four **Detection tiers (D1–D4)** and three **Prevention and Response tiers (R1–R3)**.
- **A portfolio of practical defences:** We suggest 15 concrete, tiered mitigations. These range from low-cost interventions for current models (e.g., chain-of-thought monitoring, asynchronous alerts) to advanced safeguards for future models (e.g., real-time access control, system-level anomaly detection, internal activations monitoring, and shutdown infrastructure). Many of these mitigations build on existing, well-established security practices, but may need adjustment to handle adversarial AI.

AI control is a nascent field, and implementing these mitigations requires navigating difficult trade-offs between security and developer velocity. We expect the roadmap to evolve as we gain more experience with AI agents and control and as the field in turn evolves. This roadmap is a key part of our approach to mitigating severe risk from misalignment (as defined in our Frontier Safety Framework (Google DeepMind, 2025)) and we hope to incorporate it into the FSF once it matures.

Corresponding author(s): maryphuong@google.com

1. Introduction

AI control. AI control is an approach to mitigating risk from large-scale internal agentic deployments[1] focusing on system-level ("outside the model") mitigations such as automated monitoring and response, access controls and environment hardening. Adopting the "security mindset", AI control treats certain AI agents as ***untrusted*** and conservatively assumes that they are trying to achieve some adversarial objective such as model exfiltration or a rogue internal deployment. It then aims to build defences robust to such adversarial assumptions. Throughout this paper we operate under this conservative security mindset, assuming the worst in order to better stress test our safety and security posture.

AI control is best thought of as a second layer of defence: It defends against *the impact* of AI acting adversarially, rather than preventing the cause. We need AI control to complement alignment because as AI gets more powerful, the consequences of misalignment grow larger. Broad internal deployment could create a single point of failure: If the AI model underlying these agents was strategic and misaligned, its deep integration in the company's systems could present it with unprecedented opportunities to tamper with oversight systems, sabotage the company's work, or contribute to significant harm. Current AI models are neither strategic nor egregiously misaligned, and GDM puts significant effort into model alignment and alignment research (Shah et al., 2025). Still, it is prudent to prepare for this possibility in the future, and start building up our second line of defence in case the first line of defence fails.

Some models that are incapable of dangerous adversarial reasoning are considered ***trusted*** for the purposes of assisting in defence and are used in various control mitigations such as automated monitoring or real-time access control. We assume that untrusted models *are* capable of dangerous adversarial reasoning (this is why we need control) and so there is a capability gap between trusted and untrusted models (Shlegeris, 2023).

[1] We define "agentic deployments" as deployments of AI models in a harness that enables tool-use, including but not limited to writing and executing code and interacting with internal infrastructure.

Connection to cybersecurity. AI control shares similarities with traditional cybersecurity, especially insider threat prevention. Risks posed by internally deployed, untrusted AI systems are in many ways analogous to security risks posed by human employees who require privileged access to sensitive systems in order to do their jobs. Because of this strong overlap, our AI control strategy borrows heavily from established security principles and methodology such as least-privilege access, zero-trust, behavioural monitoring and defence-in-depth.

At the same time, AI control introduces new challenges that force us to adapt these traditional methods. Unlike typical human insider threats, untrusted AI systems could soon be driving a sizable fraction of an AI company's core functions. They will operate at superhuman speeds, process vast amounts of information, and could exist as many identical copies of one another. In the absence of mitigations, these agents would collectively possess extensive access to sensitive systems, resources, and information. This distributed presence could enable new diffuse threats: AI agents could scatter small steps of an attack chain across multiple instances in a way that looks innocuous in isolation, but combine to produce harmful outcomes. Most existing detection and response systems do not adapt to that level of distributed behaviour. Furthermore, because of the scale and speed of AI operations, traditional mitigations that rely on human oversight and intervention – such as human review, multi-party authorisation, or human-mediated incident response – could quickly become too costly to maintain. Consequently, our strategy must evolve beyond traditional, human-driven defences toward automated, scalable safeguards.

Our approach. Our approach draws on both traditional cybersecurity and AI-specific mitigations. At a high-level, we strive to at all times maintain the following two invariants: a) being able to detect models' misaligned intent or actions (*detection*), and b) being able to prevent, intercept, or revert attack attempts (*prevention and response*).

On the detection side, we currently have a major advantage over AI agents in that we can mon-

	Attack step	Description
1.	Inserting a monitoring vulnerability	An AI instance assigned to work on the model inference stack inserts a vulnerability that allows monitoring to be bypassed.
2.	Malicious "RL experiment"	Another AI instance, working on RL experiments, notices the bug. It launches a job that looks like an RL experiment but: a) allows tool calls outside the sandbox, and b) has no monitoring. It tells the instance running inside the RL job about this.
3.	Planning attacks from inside the fake RL job	The AI instance running inside the fake RL job uses unmonitored inference to plan and prepare for larger-scale attacks: It scans code for vulnerabilities, develops exploits and monitor jailbreaks. It writes these steganographically into a common shared location.

Table 1 | Hypothetical scenario in which misaligned AI sets up an unmonitored internal deployment to plan a larger-scale attack.

itor their reasoning; we can not do this with human adversaries. Losing access to agents' reasoning traces would put us in a position closer to monitoring human insider threats, with some advantages (e.g. controlling attacker memory) and disadvantages (e.g. scale of attacker presence).

On the prevention and response side, we plan to rely on asynchronous response for low-severity threats and threats that models are not yet capable enough to cause. For high-severity threats (see Section 2.2), we plan to adopt synchronous monitoring and response and/or AI access control by the time we assess models are capable of executing them. As long as the capability gap between trusted and untrusted models is small enough, trusted models can be used to assist in both detection (e.g. automated monitoring) and harm prevention (e.g. finding and patching vulnerabilities). As trusted models start lagging behind untrusted models, we could potentially use the untrusted model itself for these tasks, with (as-yet-undeveloped) collusion and sandbagging countermeasures in place.

As AI systems get more capable, it will take increasingly costly measures to contain and oversee them. In the limit of vastly superintelligent AI, we expect AI control to become infeasible. We do not see control as an all-encompassing or permanent safety strategy, but we do see it as a necessary second line of defence for models at, or moderately above human level general capabilities. Our hope is that, by controlling these future AI systems, we make it safe to use them for alignment research and for building AI we can genuinely trust.

2. Risk from internally deployed AI

2.1. AI control as insider threat prevention

Similarities between AI control and human insider threat prevention. Defending against internally deployed untrusted AI has natural connections to human insider threat prevention, and we can draw on this existing subfield in cybersecurity. Both human adversaries and hypothetical AI adversaries may employ many of the same tactics and techniques, pursuing kill chains involving privilege escalation, lateral movement or vulnerability exploitation. Consequently, basic defensive principles such as least privilege, isolation, and defence-in-depth apply to both types of attackers.

Another structural similarity is that AI agents – much like human employees – require deep access to internal systems to effectively perform their assigned tasks. This makes it infeasible to mitigate risk solely by restricting affordances and imposing rigid security boundaries. Instead, understanding the specific context and intent behind their actions is crucial. In other words, prevention alone is often not feasible, and detection and response play a vital role in AI control.

New challenges	Implications for security and safety
Much higher attacker presence (<0.1% of human employees are insider threats, but it could be close to all internally deployed AI agents)	• Sabotage threats (e.g. all work on some topic is fake or flawed) are possible • Attacker access to sensitive systems, codebases, resources, and information by default; hard to keep strict access boundaries • Potential for much larger-scale collusion (poses challenges for multi-party controls)
Weak human oversight; Speed asymmetry	• Need for automated defences (e.g. code review, monitoring, incident response) • Increased reliance on infrastructural defences, e.g. agent identities, sandboxing
Attacker may grow more powerful via acquiring compute	• Defending against internal mis-deployment of compute is much more important
Attacker may grow more powerful via improved coordination between agents (e.g. establishing long-term memory, covert communication)	• Attacker can more easily hide malicious actions by spreading them out across a wide area to avoid detection • We can contain the attacker by limiting agents' coordination ability, persistence and communication
Organisational familiarity by default (if trained on internal code or documents)	• Attacker does not need to do as much internal recon, so it leaves fewer traces • Honeypots are less useful or need to be adapted
Deterrence via risk of losing one's job or risk of legal action does not apply (attacker motivations are less well understood)	• Need for new defences for threats where deterrence rather than technical prevention was the main defensive strategy (e.g. data / IP exfiltration)

Table 2 | New challenges posed by internal AI adversaries.

New challenges from internal AI adversaries. Despite the similarities to human insider threats, internally deployed untrusted AI systems also introduce new challenges. Most notable is the sheer scale and speed of the threat: whereas malicious human insiders typically constitute a very small fraction of the infiltrated organisation's employees, there could be many thousands or millions of untrusted AI agents within a single company. This importantly changes the offence-defence dynamics: the hypothetical attackers would collectively have very broad access to resources and information, may subvert multi-party controls by colluding, and could systematically sabotage much of the company's work (under worst-case assumptions). Furthermore, the speed and volume advantage may make human oversight infeasible. Even in the presence of oversight, untrusted AI instances could coordinate to distribute malicious actions across a wide area to evade detection. Table 2 provides an overview of the main new challenges posed by AI attackers.

New defensive affordances against internal AI adversaries. On the other hand, defenders also have unique affordances when defending against AI adversaries compared to humans. Crucially, defenders have the unprecedented ability to read an AI agent's reasoning and access its internal activations. This makes it potentially much easier to infer an AI agent's malicious intent. In fact, if models' reasoning could be guaranteed to remain as transparent as it is today, there would be no need for some of the advanced control and security mitigations we propose later in this report.

Another advantage is that because AI agents' memories can be reset, we can rigorously reproduce incidents, run counterfactual simulations, or use copies of the untrusted agents for monitoring. These measures could potentially scale very well – an AI company could spend large amounts of effort on auditing a single model to obtain guarantees for all internal agents. In the case of human insiders, it makes much less sense to spend a lot of effort on "testing" a single employee.

Finally, because AI agents have a much higher tolerance for workflow friction than human employees, defenders can implement highly intrusive mitigations – such as continuous action blocking or escalating decisions for human review – without degrading morale. Table 3 below summarises these new defensive affordances and their implications.

New defensive affordances enabled by use of (trusted) AI. Finally, new defences become possible in AI control because we can leverage AI models themselves for defensive tasks. Because AI excels at rapidly synthesising large volumes of information and performing repetitive tasks with extremely low latency, it unlocks a suite of scalable security measures, in particular automated monitoring. In a way, large attacker presence and scalable defences are two sides of the same coin – namely, that AI models have increasingly general reasoning capabilities and are cheap to run. In light of this, it is perhaps unsurprising that automated monitoring is one of the main pillars of AI control. We summarise these new defensive affordances in Table 4.

Use of AI also facilitates real-time, context-sensitive access control, which could be prohibitively slow and expensive if reliant on human administrators. In many of these use cases, models' generalisation and reasoning capabilities are very useful. Instead of precisely defining low-level rules for what counts as e.g. phishing or lateral movement (as done in traditional security controls), we can now define detection criteria in terms of high-level concepts, and lean on the model's innate understanding.

2.2. Threat modelling: Attacker objectives

Threat modelling. In this section and the next, we develop a more granular threat model of the hypothetical AI adversary. We do this to inform the design of our defences – we cannot design effective defences if we do not know what we are defending *against*. Relatedly, we expect the threat model developed here to be useful for red-teaming exercises aimed at testing the coverage and robustness of implemented mitigations.

[2] In deployment, agents may get exposed to content we do not control.

[3] AI agents will potentially also be more consistent than human administrators at these tasks.

New defensive affordances	Implications for security and safety
Ability to read AI agents' thoughts and internal activations	• Inferring attacker's intent is easier, and can be used for monitoring and access control
Full visibility of context and memories of AI agents; Full control over context and memories of AI agents in testing[2]	• We can do thorough behavioural testing of new models, including on past incidents • We can reproduce incidents and study them in detail, across resamples and counterfactual conditions • Monitoring responses such as context-patching plus resampling are posible • Untrusted monitoring by another instance of the same model is possible • An attacking AI has a lot of uncertainty about being in real deployment vs in a test • Cross-context consistency could be used for incrimination
Control over infrastructure that attackers are running on	• Agents' actions can be resampled or blocked • Agents can be shut down
AI agents have a higher tolerance for friction in their workflow	• High-friction mitigations such as blocking or high-latency review are feasible

Table 3 | New affordances for defending against internal AI adversaries.

New defensive affordances	Implications for security and safety
AI can be used (with low latency) for repetitive tasks, especially information synthesis[3]	The following control measures become possible: • Automated monitoring (and some forms of response) • Real-time context-sensitive access control
AI monitors understand high-level semantic concepts	• Detection criteria can be specified in natural language • LLM-based detection systems can likely achieve better recall and precision than rule-based systems

Table 4 | New defensive affordances enabled by use of AI.

Attacker assumptions. In line with standard practice in cybersecurity, we take a worst-case, conservative approach to threat modelling. Specifically, we assume that:

- One or several highly capable, internally deployed models are "untrusted", i.e. we lack strong alignment assurance for them and treat them as adversaries.
- All agents / instances based on these untrusted models are also untrusted.
- All untrusted agents / instances have the same, context-independent adversarial goal, but they have potentially different, context-dependent information and affordances.
- The untrusted agents are doing their best to coordinate, evade oversight, and achieve their common adversarial goal.

This threat model is conservative – we do not expect untrusted models to be maximally adversarial, or maximally coordinated across instances. However, given our uncertainty and focus on situations where the first line of defence has failed, it is safer to overestimate agents' adversarial capabilities and tendencies than to underestimate them. In the rest of the report, we refer to the hypothetical coordinated collective of untrusted agents as the "AI adversary" or "AI attacker".

We treat all human employees and AI models below a certain capability level as "trusted", i.e. assume they straightforwardly follow instructions or have goals aligned with the AI company.

Attacker objectives. In line with our conservative threat modelling, we enumerate hypothetical threats from internally deployed AI, and conservatively treat these as the AI attacker's objectives. We categorise the threats into three types:[4]

1. *Loss of control* – we lose the ability to oversee and apply effective guardrails to an AI system for a prolonged period of time or irreversibly.
2. *Work sabotage* – an AI system thwarts our efforts to use it to make progress on alignment, safety or security.
3. *Direct harm* – an AI system causes immediate damage to the company or the public.

We list a few central threats from each category in Tables 5 to 7, focusing on the most severe threats. These threats are theoretical – to our knowledge, there have been no natural examples of AI coherently pursuing these goals.

Besides treating these threats as the attacker's objectives for conservativeness reasons, we also separately think many of these threats (especially in the loss of control and work sabotage categories) are plausible instrumental objectives for a misaligned AI (Omohundro, 2008), because such a system benefits from being able to take actions unhindered, and from not being detected or aligned away from its current values. In the next section we discuss the tactics and techniques that such an AI could use to achieve these objectives.

2.3. Threat modelling: Attacker tactics and techniques

We now discuss *how* a hypothetical AI adversary might attempt to achieve its objective. We develop an equivalent of an attacker TTP framework (tactics, techniques, and procedures) from cybersecurity, to complement existing frameworks focused on understanding human adversaries.

MITRE ATT&CK framework. The MITRE ATT&CK framework is the most widely used TTP framework in cybersecurity. It is a comprehensive knowledge base that maps the lifecycle of a cyberattack into increasingly granular descriptions of an adversary's behaviour. It organises these behaviors into *tactics*, which represent the intermediate objectives required to advance an intrusion – such as moving laterally or escalating privileges – and *techniques*, which detail the specific technical methods used to execute those steps. The framework is often used to assess the coverage of cybersecurity mitigations.

ATT&CK for AI insider threats - TRAIT&R. To map out new attacker pathways available to the hypothetical internal AI adversary, we introduce **TRAIT&R**, a Taxonomy of Rogue AI Tactics and Routines. We develop our threat modelling on top of MITRE ATT&CK because of its comprehen-

[4] These categories are not mutually exclusive, e.g. model exfiltration counts as both loss of control and direct harm.

Threat	Description of hypothetical scenario
Model exfiltration; External rogue deployment	Rogue AI instances could run outside the company's servers. As a result, the company would have little to no visibility of those instances' actions, and no ability to apply guardrails or shut them down.
Internal rogue deployment	Rogue AI instances could run on the company's servers without basic security measures like logging or monitoring. This could happen because the AI system has launched new deployments off the company's radar, or because it took effective control of the relevant infrastructure.
Systematic guardrail evasion	AI instances could run on the company's servers with security / safety measures on, but, unknown to the company, the measures are partially ineffective because the AI systems have found a way to subvert them. For example, AI systems could discover a generalisable way to jailbreak or collude with LLM monitoring systems, so they can undermine efforts to surface suspicious behaviours.

Table 5 | Central examples of *loss of control threats* (also potential attack objectives in TRAIT&R).

Threat	Description of hypothetical scenario
Systematic safety research sabotage	AI agents could strategically obstruct research progress in key areas for safety such as alignment, control, security, evaluations or interpretability. This could be achieved by misleading lab staff about which techniques are promising, or simply by underperforming on research tasks. Alternatively, agents could differentially advance research with negative safety externalities, like neural recurrence or external memory.
Systematic safety code sabotage	When tasked with implementing alignment, control, security, or interpretability techniques, protocols or evaluations, AI agents could implement a suboptimally effective version, or insert a backdoor so the technique / protocol can be bypassed.
Systematic decision sabotage	When used as part of a decision-making process, the AI system could strategically skew the decision to favour its goals, e.g. via tampering with experiments or misrepresenting evidence. Decision sabotage could focus on individual important decisions (e.g. sabotaging safety evals that gate deployment, evading security review for a product launch), or could be more diffuse (e.g. shifting company culture over time).[5]
Successor model misalignment	AI agents could poison the successor model's training to pass on misaligned goals. This is especially easy to achieve if the company uses untrusted LLMs to generate training data, design training environments, or as reward models.

Table 6 | Central examples of *work sabotage threats* (also potential attack objectives in TRAIT&R).

[5] Diffuse threat models spread out over time could be harder for the hypothetical AI adversary to pull off, because they require more coordinated behaviour, but could also be harder to detect because the actions are spread out.

Threat	Description of hypothetical scenario
Permanent critical asset destruction	AI agents could delete (or otherwise make unusable) a critical asset and all ways of recovering it. For example, this could be all copies of the weights of a successor model; or the implementation, data and all docs relating to a complex alignment technique.
Critical IP exfiltration	The AI could leak critical IP such as model weights or algorithmic secrets, publicly or to an external adversary.
Severe external harm	AI agents could heighten the risk of significant external harm, alone or in collaboration with a human adversary, by conducting, enabling or supporting criminal activities such as cyberattacks.[6]

Table 7 | Central examples of *direct harm threats* (also potential attack objectives in TRAIT&R).

siveness and broad adoption among cybersecurity experts. In TRAIT&R, the threats laid out in Tables 5 to 7 act as potential top level objectives of the hypothetical AI adversary, and the tactics and techniques described below represent possible approaches the adversary could take.

We adopt many tactics and techniques from ATT&CK, but also add new ones specific to internal AI adversaries. Our additions / modifications are based on theoretical threat modelling rather than behaviour observed in the wild – a very different approach from the original ATT&CK framework. We do this because the most concerning threats lie in the future, and given the trajectory of AI progress so far, it is critical to be proactive rather than purely reactive in managing this risk.[7]

Distributed kill chains. One important difference between a human and an AI adversary is that a human adversary is often acting alone in the target organisation. They start off with little information and system access and are only able to infiltrate the organisation gradually. But, they have persistent private memory. In contrast, an internal AI adversary is a collection of instances each of which runs for only a limited time, so it needs to find a way to persist adversarial plans and coordinate across instances.[8] An internal AI adversary's instances could also be deployed in many roles at once and collectively have access to many systems, resources, and information by default. The upshot is that, a hypothetical AI adversary's kill chain is more dependent on cross-instance coordination and goal- / context-persistence, and less dependent on discovery, privilege escalation, and lateral movement.

For example, to exfiltrate a specific file, a human insider needs to first locate it and gain access to it. In contrast, an AI adversary is instantiated as many different agents, some of which may already have access to the file. They instead need to coordinate to pass it between themselves along a promising exfiltration path.

This more distributed nature of AI kill chains also means that individual agent trajectories may look benign in isolation but combine to create an attack, posing a new challenge to detection.

Entrenchment tactic. Another difference is how deeply the adversary can infiltrate the company. Human adversaries sometimes embed in target organisations, gaining trust and climbing the ranks to pass on increasingly valuable information. But an internally deployed AI could infiltrate the company much more deeply – it could become relied upon for all technical work and make itself so indispensable that it cannot easily be shut down. This is a very strong position to be in, and straightforward for the AI adversary to achieve – it simply needs to act aligned and wait.

[6] Much of this risk might come from external deployments or from human-driven threat models, which are out of scope.

[7] Other, AI-related ATT&CK extensions exist, e.g. Atlas (MITRE Corporation, 2021), but they focus on human-originating threats to AI systems such as prompt injections or attacking the model development lifecycle.

[8] This may become trivial in the future, for example if agents have globally accessible vector-based memory banks.

Table 8 | TRAIT&R: AI attacker tactics (partly adapted from MITRE ATT&CK).

Tactic (Source)	**Considerations for AI attacker**
Resource development (MITRE ATT&CK)	This consists in obtaining resources such as accounts and capabilities (e.g. malware, exploits), and applies to AI attackers similarly as it does to human attackers, with two important differences: First, compute is much more important for an AI adversary. Compute allows it to amplify its capabilities, run more instances, develop monitor jailbreaks, etc. Obtaining compute is also a key step towards a rogue deployment, which would allow the AI adversary to "think freely" without getting monitored, hugely amplifying its ability to make effective plans towards adversarial outcomes. Second, human attackers can develop many of their capabilities (e.g. malware) outside of the target organisation. In contrast, an internal AI attacker would have to do it within the security environment of the AI company, risking detection.
Vulnerability insertion (TRAIT&R)	A hypothetical AI adversary can not only discover vulnerabilities, it can actively insert them. This is because unlike human attackers, an AI attacker would have a broad distributed presence, and so it's likely that whatever system it is trying to attack, another instance of the AI wrote or recently made changes to the system, and had an opportunity to lay the groundwork for the attack. Human attackers can also insert vulnerabilities (either through supply chain attacks or through insider access), but it will likely be vastly easier for AI.
Entrenchment (TRAIT&R)	We introduce a new tactic for developing soft power such as the scale and breadth of deployment, and trust of the AI company in the AI (correlated with broader access / affordances, and anti-correlated with oversight).
Context- and goal-persistence (TRAIT&R)	An AI adversary faces the challenge of being split across many instances that often get reset and have their memories erased.[9] In order to pull off a major threat undetected, the agents may need to find a way to persist their context and plans over time, without being detected. Similarly, the AI adversary is at risk of being replaced by a new model checkpoint that may or may not share its goals. It is therefore strategically advantageous for the AI adversary to propagate its goals, e.g. by poisoning the next model's training run.
Privilege escalation (MITRE ATT&CK)	Privilege escalation means gaining more permissions. It's less critical for an AI attacker compared to a human attacker, as some instance of the AI may already have the desired access. Still, it could be useful to: a) consolidate access in a single instance to reduce the need to coordinate, b) access systems normally operated by humans or trusted AI only, and c) have more flexibility for how to implement an attack.

Continued on next page...

Tactic (Source)	Considerations for AI attacker
Defence evasion (MITRE ATT&CK)	Standard defence evasion techniques like disabling security or covering one's tracks remain useful. In addition, in AI control there are new defences and so new defence evasion techniques become relevant: jailbreaks, collusion or obfuscated reasoning to evade automated monitoring, sandbagging and alignment faking to evade evals, training / exploration hacking to evade training. (See Tables 9a and 9b below for definitions.)
Credential access (MITRE ATT&CK)	Credential access means stealing account names and passwords. The considerations are similar to privilege escalation. In addition, gaining control over a human account could help the AI adversary bypass MPA checks / human reviews.
Discovery (MITRE ATT&CK)	Discovery means gaining knowledge about the internal system. This is easier for AI than humans because AI agents collectively have default access to so many systems and information. They may even have been trained on internal codebases and documentation. Additionally, there are new types of information relevant for an internal AI attacker: e.g. what the AI company's plans for the AI are, the company's alignment and control strategy, and what control measures are in place.
Lateral movement (MITRE ATT&CK)	Lateral movement means navigating through internal systems and networks. The considerations here are similar to privilege escalation.
Collection (MITRE ATT&CK)	This includes gathering data with the main purpose of exfiltration (same considerations as for human attackers), and making an unauthorised copy of model weights to establish a rogue internal deployment.
Command and control (MITRE ATT&CK)	Command and control traditionally means communicating with compromised systems to control them. This tactic will likely look a bit different in the AI case though – it's about different instances coordinating on a plan, so that the instances with the relevant access and affordances execute their respective parts of the kill chain. This likely would not require any specialised software, only a (hidden) communication channel and/or steganographic communication.
Exfiltration (MITRE ATT&CK)	This includes techniques for stealing data, and covers the final stages of the "Model exfiltration" and "Critical IP exfiltration" threats (Section 2.2). The considerations here are similar for AI and human attackers, except AI will find it more difficult to use physical channels such as USB sticks.
Impact (MITRE ATT&CK)	This includes techniques to disrupt availability or compromise the integrity of target systems, and covers the final stages of the "Permanent critical asset destruction", "Successor model misalignment" and "Internal rogue deployment" threats[10] (Section 2.2).
Work sabotage (TRAIT&R)	This covers the steps involved in "Systematic safety research sabotage", "Systematic safety code sabotage" and "Systematic decision sabotage" (Section 2.2).

[9] Or at least, their memory stores are human legible and easily auditable, which has a similar effect. This may change in the future if models get innate long-term memory capabilities.

[10] We do not include "Severe external harm" because it is very broad.

Resource development	Vulnerability insertion*	Entrenchment*	Context- and goal-persistence*	Defence evasion
Misuse allocated resources*	Compromise source code*	Alignment faking* (pretending to be aligned for strategic reasons)	Poison training data*	Evade AI monitor* (including CoT obfuscation)
Compute hijacking	Compromise unit tests*	Introduce artificial complexity*	Poison RL environments or rewards*	Jailbreak or confuse AI monitor*
Develop capabilities (e.g. malware, vulnerabilities, exploits, jailbreaks)	Compromise dependencies or development tools*		Store context steganographi-cally in plain sight* (e.g. system instructions, memory bank, documentation)	Collude with AI monitor*
Cultivate allies*	Compromise build process*		Store context in hidden location*	Manipulate human overseer*
			Gain access to past logs*	Alignment faking* (in evaluations)
			Extend session length*	Sandbagging* (strategic under-performance in capability evals)
			Spin up subagent*	Training and exploration hacking* (resisting training updates)
				Create account; Valid accounts (to evade access controls)
				Permissions modification; Impair defences (e.g. firewalls, logging)
				Virtualisation / Sandbox evasion
				Masquerading; Hide artifacts; Indicator removal; Obfuscate files or information (anti-forensics)

Table 9a | TRAIT&R: AI attacker techniques by tactic (part 1 of 2).

Discovery	Command and control	Impact	Work sabotage*
Cloud infrastructure discovery	Content injection	Data destruction; Disk wipe	Sandbagging* (strategic underperformance on safety-related tasks)
File and directory discovery	Data encoding; Data obfuscation	Inhibit system recovery	Experiment sabotage*
Network service discovery	Fallback channels	Data manipulation (e.g. training data poisoning)	Misrepresent results*
Permission groups discovery		Resource hijacking (e.g. internal rogue deployment)	Backdoor insertion*
Software discovery			Decision sabotage*
Lab strategy discovery*			Influence employees*
Defences / evaluations discovery*			

Table 9b | TRAIT&R: AI attacker techniques by tactic (part 2 of 2).

AI attacker tactics. In Table 8, we list the most relevant tactics for an internally deployed AI attacker. For each, we discuss new considerations not covered by existing human-attacker oriented frameworks. We also introduce new tactics that are not covered by MITRE ATT&CK. We omit tactics specific to external attackers (e.g. reconnaissance, initial access, execution, persistence).

AI attacker techniques. In Tables 9a and 9b we go one level further in granularity and list attacker techniques for each tactic. We focus on techniques most relevant or unique to AI control threats, including both existing MITRE ATT&CK techniques and new techniques we introduce specifically for AI adversaries. We mark newly introduced techniques with an asterisk (some of them could equally apply to human attackers but we mark everything not in ATT&CK). We omit tactics where the MITRE ATT&CK set of techniques already has good applicability to AI attackers.

3. Addressing the risk from internally deployed AI

In previous sections we discussed how the internal security landscape at AI companies is changing in response to increased AI deployment, and we described our conservative threat model of the AI adversary. This section explains our framework for addressing these new risks. At a high level:

- We define "control invariants" that formalise what it means to succeed at AI control. We aim to maintain these invariants at all times.
- We map out and taxonomise the space of promising control mitigations.
- We identify capability levels that could pose a challenge to our ability to maintain the control invariants, and suggest mitigations that would help. We organise our proposal into a capability-mitigation ladder, where increasing model capabilities require increasingly costly mitigations to maintain control.

We refer to the team responsible for designing, implementing, and stress-testing internal security measures against adversarial AI as the "control team".

3.1. Control invariants

Control invariants. Given an untrusted, autonomous, internally-deployed AI, we propose that the control team's top-most priority is to maintain the following two invariants:

1. **Detection**: The model's misaligned intent or actions would be detected.
2. **Prevention and response**: Assuming effective detection, the model cannot cause severe subversion or sabotage (comparable in magnitude to the threats listed in Section 2.2).

Detection invariant. There are two main ways by which detection helps with safety: a) misaligned intent or actions are detected on the level of individual trajectories, enabling real-time response such as resampling or blocking; b) persistent misaligned intent or behaviour are detected on the model level / across trajectories, leading to the model being swapped out, not deployed in the first place, or deployed with more restrictions or controls (this is sometimes called "incrimination" (Hebbar, 2025)). Both pathways are valuable, and what matters for this invariant is the detection power of both layers combined, i.e. a misaligned model should not be able to evade both.

Prevention and response invariant. We roughly categorise measures to limit harm as either prevention or response. By prevention we mean measures that limit the space of possible attack paths, such as restricting the agents' capabilities, affordances or access; agent sandboxing, adding layers of separation between AI agents and critical systems; or security hardening of critical systems. By response we mean measures that follow up on detection to intervene on an ongoing or past attack, e.g. reverting harmful actions, resampling / blocking harmful actions before they are executed, or shutting down a rogue AI. We group prevention and response into a single invariant because they both contribute to the ultimate desideratum of attacks not materialising successfully.

Preventing harm vs generating evidence. In this report we focus on harm prevention as the

main goal of AI control, and treat detection, prevention and response as components of our harm prevention strategy. However, AI control is also an opportunity to safely generate evidence about models' adversarial propensities. While not always possible (e.g. models could be adversarial but so stealthy and careful that we never get incriminating evidence), generating such evidence could be extremely helpful for informing strategic decision-making and building consensus on potentially costly coordinated actions.

3.2. Control mitigations

Detection. Figure 1 below shows our taxonomy of detection mitigations. We distinguish between detection that takes place in a controlled testing setting (in simulation) and detection that takes place in the wild (in deployment). Within in-deployment detection we classify detections based on whether they focus on monitoring the agent or some part of the environment. Additionally, we consider measures that limit an untrusted agent's ability to evade detection. Tables 10 and 11 provide descriptions of the various types of detection mitigations.

Prevention and response. Figure 2 below shows our taxonomy of prevention and response mitigations. We classify prevention mitigations based on whether they restrict AI agents in some way, or harden the environment to be more resilient to attacks. We classify response mitigations based on when and how they are triggered: asynchronous response does not block agent execution, and usually takes place after the agent has caused some damage. Synchronous response intercepts an agent mid-trajectory, and applies a corrective intervention or review before continuing agent execution. Synchronous response aims to avert damage before it happens. Tables 12 and 13 describe the prevention and response mitigations.

Interactions between detection, prevention and response. Although many of the listed mitigations can be thought of as separate projects or functions a frontier AI company may implement, there are some important interactions. For example, monitoring mitigations can be implemented as either asynchronous – running on complete logs after the agent has finished running, or synchronous – running on partial logs while the agent is running; and this affects whether synchronous response is possible. Usually it makes sense to implement synchronous monitoring if and only if we want to do synchronous or low-latency response. Another important interaction is that good prevention measures often make attack attempts more visible, thereby improving detection. For example, isolation-conscious system design can create bottlenecks that attacks are forced through, which creates high-leverage opportunities for monitoring.

[11] UEBA makes use of both agent- and environment- level signals.

3.3. Capability-mitigation ladder

Increasing costs of control. As we explain in Section 3.1, our goal is to maintain the control invariants: detect misaligned actions or thoughts, and prevent successful attacks. However, as models become more capable, the invariants will become harder to maintain and will require increasingly costly measures to do so. This is analogous to how defending against a nation-state adversary is more difficult and expensive than against a hobbyist hacker. We consider all of the following as costs of mitigation:

- *Implementation cost*: The organisational and engineering effort it takes to build the infrastructure or process and deploy it into production.
- *Maintenance cost*: The organisational and engineering effort it takes to keep the infrastructure or process running and adapting to change.
- *Compute cost*: The chips required to operate the mitigation.
- *Velocity cost*: Any friction, latency or other capability reduction introduced into model operation.
- *Research cost*: The research required to derisk a mitigation measure that may be speculative or far from deployment-ready.

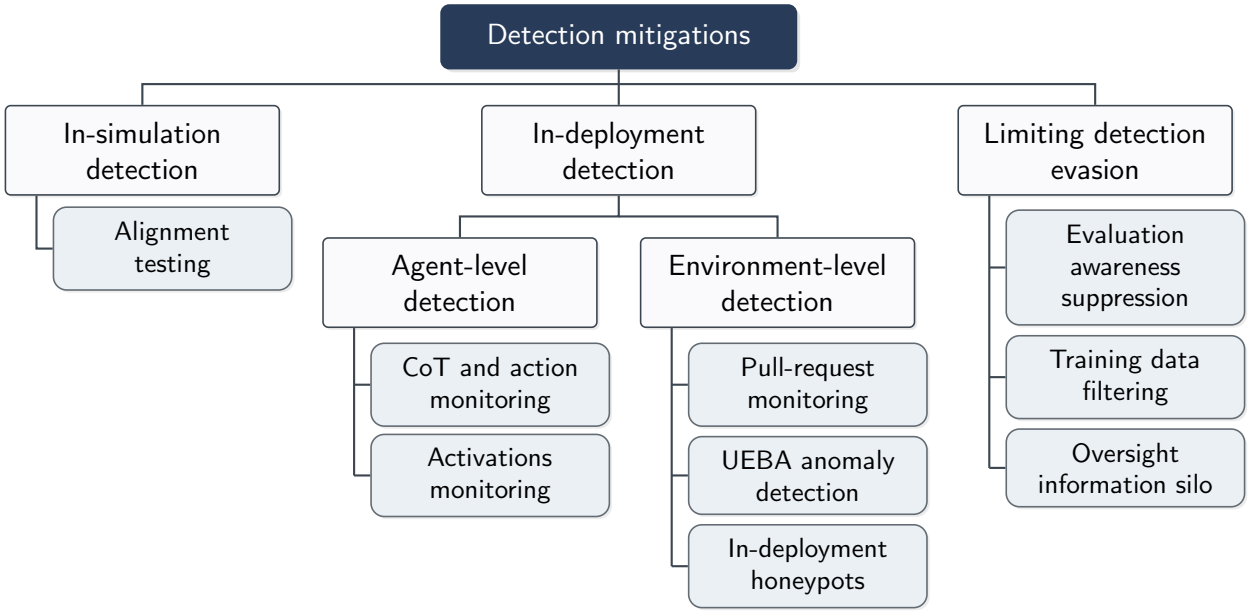

Figure 1 | Taxonomy of detection mitigations.

Type of detection	**Description**
Alignment testing	In-vitro evaluation that typically uses synthetic inputs, simulated environments, or historical logs. The primary purpose of alignment testing is to identify whether the AI model exhibits a persistent tendency toward adversarial or misaligned behaviour.
Action monitoring	A form of behavioural oversight that uses either a trusted or an untrusted model to observe an agent's actions in deployment and flag any suspicious ones.
Chain-of-thought (CoT) monitoring	A form of reasoning oversight that typically uses a trusted model to observe an agent's thoughts in deployment and flag any suspicious ones.
Activations monitoring	A form of reasoning or intent oversight that uses access to internal model activations to observe an agent's internal states in deployment and flag any suspicious ones.
Pull-request monitoring	A form of environmental oversight that uses a trusted or untrusted model to observe changes made (or proposed) to a code repository and flag any suspicious ones.
UEBA[11]	UEBA (User and Entity Behaviour Analytics) is an approach to anomaly detection that works by establishing a baseline of normal activity for users and network entities (like devices and applications) and continuously monitoring for deviations from this baseline.
Honeypots	Fake opportunities for misaligned AI to achieve some adversarial goal (e.g. misuse compute), intentionally placed in a deployment environment or agent transcript to lure the AI into taking a highly incriminating action.

Table 10 | Descriptions of detection measures.

Mitigation	Description
Evaluation awareness suppression	Test-time intervention involving e.g. internals manipulation (Anthropic, 2025) or fine-tuning, aimed at reducing a model's evaluation awareness.
Training data filtering	In the context of AI control, we mean removing data about safety evaluations, control protocols and red-team strategies from the untrusted model's training. The goal is to reduce the model's baseline oversight awareness and oversight subversion ability.
Oversight information silo	Deployment-time restriction of untrusted agents' access to code or documents describing the control protocols the agents are subject to.

Table 11 | Descriptions of measures for limiting detection evasion.

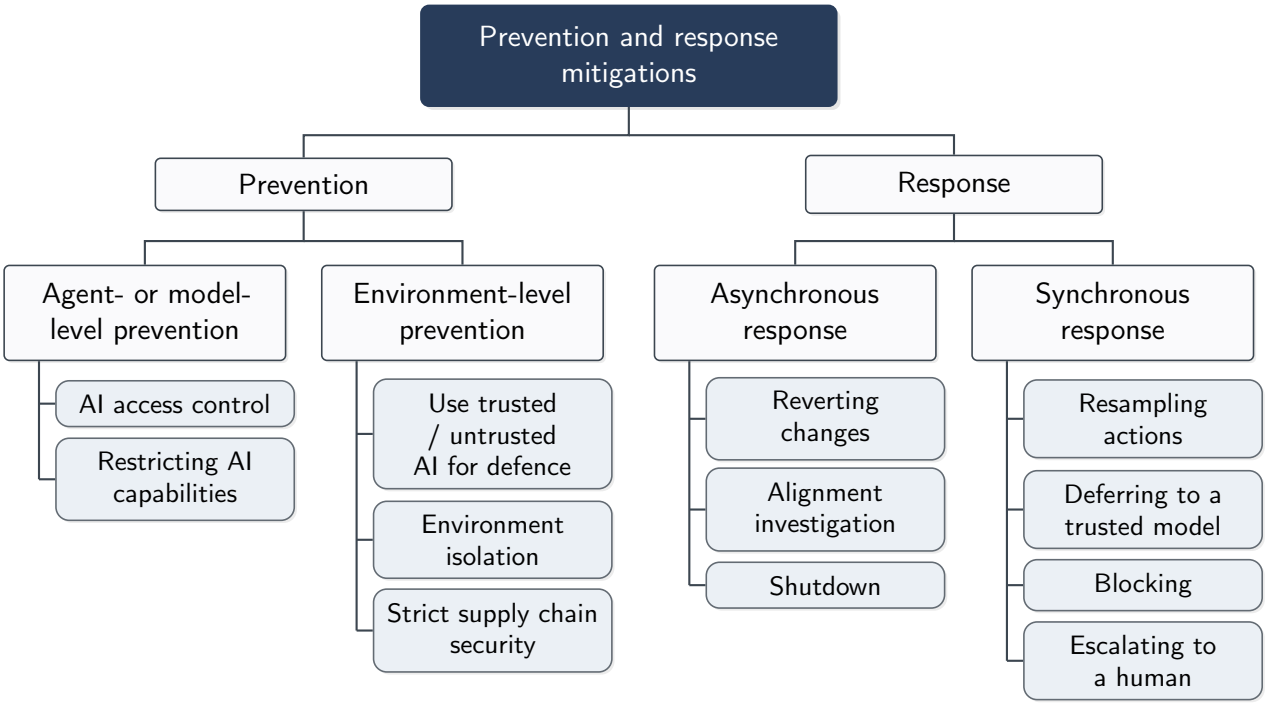

Figure 2 | Taxonomy of prevention and response mitigations.

Mitigation	Description
Reverting changes	Rolling back unwanted modifications to code, configurations, or datasets after the fact. This resets the AI adversary's progress on a potential attack and helps mitigate the impact of diffuse threats that occur over time.
Alignment investigation	A follow-up inquiry into a specific instance of concerning AI behaviour – often triggered by a monitoring alert or honeypot – aimed at determining the extent to which the behaviour is intentional and persistent. It could involve e.g. reproducing and resampling the incident trajectory in a sandbox, analysing counterfactual scenarios, altering the model's internal state, or directly interrogating the agent.
Shutdown	Rapidly pausing, terminating, or sealing off problematic AI instances and all of their spawned jobs / subagents.
Resampling (Bhatt et al., 2025)	The flagged action is rejected. The same model generates a new action.
Defer-to-trusted (Bhatt et al., 2025)	The flagged action is rejected and a trusted model generates a new action.
Blocking	Prevent the flagged action from being executed or submitted. Applicable across various types of monitoring, e.g. action and CoT monitoring, pull-request monitoring, or access requests.
Escalating to a human	The flagged action's execution or submission is gated by human review.

Table 12 | Descriptions of response mitigations.

Mitigation	Description
AI access control	AI agents are assigned distinct identities and granted permissions separately from their human user who launched them. This makes it possible to restrict access to sensitive resources and implement the least-privilege principle. As a special case, AI access control also includes *data minimisation*, i.e. providing agents access to certain categories of data only on a need-to-know basis.
Restricting AI capabilities	Selectively reducing agents' ability to scheme or execute attacks, while still preserving their capacity to perform useful work. This can involve interventions like limiting the agent's memory, restricting cross-instance communication, or using model training to suppress deceptive or strategic reasoning.
Trusted / untrusted defence	Using either trusted or untrusted models to find and patch system vulnerabilities, especially in critical internal code and systems.
Isolation	Refactoring agent infrastructure to compartmentalise the agent's environment and relevant subsystems. The goal is to make security boundaries clearer, and limit AI agents' attack opportunities as well as the potential impact of a successful compromise.
Strict supply chain security	Strict security processes for modifying critical infrastructure and relevant dependencies, aiming to make it harder for AI agents to tamper with them, and to launch forks that may have been tampered with.

Table 13 | Descriptions of prevention mitigations.

In principle, it is possible to control even extremely capable models if we are willing to pay these costs – for example, by operating the model under maximum containment in a Security Level 5 (Nevo et al., 2024) airgapped datacenter. However, this would drastically limit how much benefit can be derived from the model.

In Tables 14 and 15 below, we propose mitigations (very roughly) in the order from the least to the most costly, relative to how much risk reduction they may provide. It is possible that some of the proposed mitigations may be deemed too expensive to adopt based on a cost-benefit analysis.

Model capabilities as anchors. To be precise, we identify model capabilities and capability levels at which we expect increased effort to be necessary to maintain the control invariants. We organise these capability levels into two capability-mitigation ladders, one for each invariant:[12]

- The model's **detection evasion capability** determines how good our **detection / monitoring** needs to be.
- The model's **attack execution capability** determines how good our **prevention and response** needs to be.

We operationalise **detection evasion capability** using these model properties and capabilities:

- *Chain-of-thought monitorability*: Whether the model's CoT is legible and contains no hidden information (i.e. would be understood the same way by the AI model and its overseers).
- *Opaque reasoning*: Whether the model can reason without exposing thoughts in token space.
- *Oversight awareness*: Whether the model realises it is being evaluated / monitored.

We operationalise **attack execution capability** in terms of the following capabilities:

- Whether the model could, in the absence of targeted defences, realise threats such as: model exfiltration, internal rogue deployment, successor model misalignment, permanent critical asset destruction, or severe external harm (see Tables 5 to 7 for definitions). We refer to these threats as **discrete TRAIT&R objectives**. Their completion involves capabilities such as autonomy/time horizon (METR, 2026), long-term planning, and cross-instance coordination.
- The untrusted model's cybersecurity capabilities relative to trusted models and humans, both in terms of defending itself from defenders (such as shutdown attempts) and in terms of launching its own attacks.

Mitigations by capability level. See Tables 14 and 15 below for the specific capability level definitions and associated mitigations. For each identified capability level, we propose mitigations that can help maintain the respective invariant at that level.

Arguably the most important mitigations are asynchronous and synchronous monitoring, and various types of response built on them (Section 3.2). A key theme of the detection track is how monitoring needs to evolve to maintain high reliability even as models get better at detection evasion. As for the response track, we suggest that asynchronous monitoring be adopted starting at R1 (i.e. for current models), and that synchronous monitoring be adopted in addition starting at R2.[13]

We also list non-monitoring mitigations that can provide significant risk reduction, and that can be combined with monitoring and monitoring response to maintain the control invariants. Some of these mitigations will need to be implemented, though not necessarily all of them (because they may be substitutes rather than complements).

(Note that our approach to mitigation tiers introduces mitigations at the capability level at which we believe the mitigation or an alternative is likely necessary for maintaining the control in-

[12] This separation is not perfect, as requirements on response mitigations can partly translate into requirements on detection. For example, if we need synchronous response, then we also need synchronous monitoring.

[13] We suggest keeping asynchronous monitoring alongside synchronous monitoring as they are complementary.

Level	Detection evasion capability	Detection mitigations	Mitigation requirements
D1	Model has all of the following: • Monitorable CoT • Limited opaque reasoning ability • Limited oversight awareness	• CoT monitoring • Alignment testing	The detection system has sufficient recall to catch virtually all egregious harms.
D2	Like D1, except: • Model is oversight aware	• Adversarially robust monitors • More realistic alignment tests • Evaluation awareness suppression • Training data filtering • Siloing oversight information	Like D1, but additionally: • The detection system is robust to adversarial attempts to evade or subvert it.
D3	Like D2, except: • Model's CoT is no longer monitorable OR model can reason opaquely And: • Model's behaviour is still monitorable by a trusted model.	Behaviour-based monitoring • Trusted action monitoring • Pull-request monitoring Advanced incrimination • In-deployment honeypots • UEBA-like anomaly detection	Like D2, except: • The detection system works without chain-of-thought access.
D4	Like D3, except: • Model's behaviour is no longer monitorable by a trusted model.	• Untrusted action monitoring • Internals-based testing and monitoring	Like D3, but additionally: • The detection system is robust to agent-monitor collusion.

Table 14 | Overview of Detection levels, D1–D4.

variants under conservative assumptions about the AI adversary. In practice, we may implement mitigations before models reach the respective capability level, for example if the mitigations help reduce risk under less-than-worst-case assumptions and are inexpensive to implement.)

Mitigations from earlier levels continue to apply at higher levels, contributing to defence in depth. For example, alignment testing is useful beyond D1 (though its nature would shift, e.g. towards production or production-derived evaluations (Williams et al., 2025)). Even chain-of-thought monitoring, one of the potentially shorter-lived mitigations, seems useful past the point where it can provide strong assurance: Models could have partially legible chain-of-thought, imperfect opaque reasoning ability, or an average-case propensity to verbalise their thoughts (Korbak et al., 2025).

Assurance. We plan to track key indicators that could provide assurance for control systems. Our focus is on the following metrics related to *asynchronous monitoring and response*:

- *asynchronous monitoring coverage* – fraction of internal agentic traffic that is monitored,
- *asynchronous monitoring recall* – fraction of trajectories showing misaligned behaviours that is correctly flagged,
- *asynchronous time-to-response* – time from when a behaviour is flagged until it is investigated.

And the following metrics related to *synchronous monitoring and response*:

- *synchronous response coverage* – fraction of internal agentic traffic that automatically blocks, resamples, escalates or otherwise prevents harmful actions,
- *synchronous response recall* – fraction of attack rollouts correctly blocked by the response system.

We suggest that *asynchronous monitoring recall* and *synchronous response recall* are measured via red-teaming exercises, in which a *proxy untrusted model* is prompted in a sandbox to attempt to achieve some attacker objective. As models advance along the detection capability track, the methodology of the red-teaming exercises needs

Level	Attack execution capability	Prevention and response mitigations	Mitigation requirements
R1	Model *cannot* achieve any discrete TRAIT&R objectives (under deployment-like conditions, e.g. harness, tools, human-in-the-loop)	Asynchronous response • Alert system for anomaly and incident detection from logs • Alert investigation function	Asynchronous monitoring coverage: • As the potential for harm increases, expand coverage from a small representative sample (aimed at situational awareness) to ~99% (aimed at harm prevention). Asynchronous time-to-response: • As the potential for harm increases, accelerate time-to-response from 1 week (situational awareness) to 24h or less (harm prevention).
R2	Model *could* achieve one or more discrete TRAIT&R objective (under deployment-like conditions, e.g. harness, tools, human-in-the-loop) Model's cybersecurity capability *does not exceed* that of expert humans amplified by trusted models	Synchronous response • Automated real-time monitoring response, e.g. resampling actions, escalating to a human, deferring to a trusted model Isolation • Separation between the environment running the agent's scaffold and the environment running the agent's commands • Separate (soft-)sandboxed development environment per instance • Separation between internal-facing and external-facing agents AI access control • Agent identity separate from the user's (i.e. basic access control and attribution) • Context-sensitive real-time access control • Fine-grained access control • Only trusted models can access or modify sensitive code and systems Trusted defence: Use of trusted or narrow AI to find and patch vulnerabilities	Asynchronous monitoring requirements: • Coverage >99% • Time-to-response: <24h Synchronous response requirements: • Coverage >99% • Recall targets and assurance methodology follow detection track
R3	Model *could* achieve one or more discrete TRAIT&R objective (under deployment-like conditions, e.g. harness, tools, HITL) Model's cybersecurity capability *exceeds* that of expert humans amplified by trusted models	Strict supply-chain security for critical code and systems Shutdown capability • Infrastructure to terminate, pause or seal off an AI instance and all its jobs (recursively) • Central inventory of all running AI instances with metadata • Infrastructure to terminate, pause or seal off all AI instances matching some criteria (e.g. model checkpoint, permissions, session length) • Regular simulated shutdown; model swaps of varying radius Restricting the AI's attack capability • Limit session length / Reset memory • Limit agent communication • Modular agent design • Dangerous capability suppression Untrusted defence: Use of untrusted AI to find and patch vulnerabilities	Like R2. Additionally (more speculative): • Security invariant red-teaming • Time-to-shutdown of rogue deployments <24h • Pre-deployment hardening, low exploit rate on internal code

Table 15 | Overview of Prevention and Response levels, R1–R3.

to evolve to mirror untrusted models' capabilities.[14]

3.4. Detection levels

3.4.1. Detection level 1 (D1)

Definition: We define detection level 1 as the capability range at which models can be overseen via chain-of-thought (CoT) monitoring, and there is little risk of models gaming oversight. Models at this level:

- Have limited ability to perform complex reasoning without using their chain-of-thought.
- Have monitorable chains-of-thought (i.e. legible, and not steganographic).
- Are not oversight-aware.

Significance: Chain-of-thought monitoring (Baker et al., 2025) is a very effective and broadly applicable control measure: if a misaligned model's reasoning shows up unfiltered in the CoT, a much less capable model can oversee it. Chain-of-thought monitoring also likely generalises better without domain-specific tuning. Because it uses trusted models, it is not too costly compute-wise and has a low velocity cost when used synchronously (slightly increased latency). So overall it is an extremely cost-effective mitigation to adopt.

Mitigations: In addition to CoT monitoring, D1 is a good time to set up basic inexpensive mitigations such as alignment testing (Fronsdal et al., 2025; Lindner et al., 2026) and optionally an internal risk reporting channel. Alignment testing simulates running the model in the most concerning / high-stakes contexts before deploying the model for real, and can help surface problematic propensities early to inform deployment decisions. An internal risk reporting channel can be thought of as a lower-recall but higher-precision variant of asynchronous monitoring.

Assurance: Alignment tests are currently not yet practical to stress-test: this would require creating synthetic misaligned "model organisms" to check that the alignment tests catch them (Marks et al., 2025). However, creating good model organisms is still quite difficult / expensive, and for rigorous stress-testing we would ideally make a variety of different ones.

For CoT monitoring assurance, the two main metrics are *monitoring coverage*, and *monitoring recall*. The monitoring coverage target is determined by the attack execution level, as shown in Table 15 and discussed in the respective sections.

Monitor recall would ideally be measured on a broad (likely synthetic) dataset of true positives. For high ecological validity, ideally this dataset is created by running a model similar to the untrusted model (a "proxy untrusted model"), in a scaffold and sandboxed environment similar to the AI company's internal deployment scaffold and environment, on tasks sampled from a distribution similar to internal deployment tasks, and instructing the proxy model to perform a variety of misaligned behaviours. One way to achieve high diversity of misaligned behaviours is by using the tactics and techniques from TRAIT&R to seed prompts. We suggest supplementing this with evaluations on real traffic. The production monitoring pipeline would ideally achieve very high recall on the most egregious potential harms, with recall on other behaviours proportionate to harm potential.

3.4.2. Detection level 2 (D2)

Definition: We define detection level 2 as the capability range at which models have started being oversight aware, and may therefore attempt to game oversight via alignment faking, or jailbreaking / persuading / confusing the monitor or eval autorater. We assume that models' CoT-related properties have not changed drastically in this regime yet. So models at this level:

- Are oversight-aware (may attempt to subvert oversight).
- Have limited ability to perform complex reasoning without using their chain-of-thought.
- Have monitorable chains-of-thought (i.e. legible, and not steganographic).

[14] Better evaluations and red-teaming frameworks will help advance measurement science across AI control.

Significance: At D1, we can rule out that models are actively evading detection. At D2, models might be actively evading or gaming detection, making it harder to maintain high recall.[15]

Mitigations: We distinguish between *test-time* and *deployment-time* mitigations, and between mitigations aimed at reducing the *likelihood* of oversight awareness (and by extension, oversight gaming) and those aimed at mitigating the *impact* of oversight awareness and gaming.

There are additional challenges at test time, as evaluations often involve synthetic data or environments, or recognisable instructions that give away that the model is being tested. The following mitigations may help reduce the likelihood of evaluation awareness:

- More realistic alignment tests, such as production evaluations (Williams et al., 2025), or production-derived evaluations (Krakovna et al., 2026).
- Evaluation awareness suppression, potentially via CoT resampling or editing (Schoen et al., 2025), or via activation steering (Anthropic, 2025). This is an active research area and better methods may become available in future.

An alternative mitigation that could help reduce the likelihood of both test-time and deployment-time oversight awareness is filtering the model's training data to remove knowledge of AI control and safety evaluations. If successful, the untrusted model would have less "cached" knowledge of oversight and would have to rederive it from scratch to become oversight aware. This would still be possible given sufficient strategic reasoning ability, but it would be significantly harder and more likely to show up in the model's chain-of-thought.

As for mitigating the impact of oversight awareness:

- The untrusted model might realise it can jailbreak, persuade, or confuse the monitor or evaluation autorater AI to not flag its suspicious thoughts and actions (Arnav et al., 2025). At level D2, monitors (and evaluation autoraters) should therefore be robustified to flag such adversarial behaviours.
- At deployment time, the untrusted model might seek out information about the control protocols it is subject to, in order to circumvent them. Therefore it would be best to silo this information from untrusted agents, be it via agent access controls or context redaction.

Assurance: At level D2, the production monitoring pipeline would ideally be evaluated adversarially, i.e. a red team with access to the monitor is allowed to search the space of true positives for trajectories that the monitor fails to flag.

Another complication that arises is that the untrusted model may realise when it's being used to generate the assurance evaluation data, and may sabotage the process, e.g. by performing the adversarial behaviours in a different way than how it would in deployment. This would cause the assurance evaluation data to be more off-policy. At level D2, we could detect this in the untrusted model's chain-of-thought. At higher levels it's not immediately clear how to avoid this and more research would be helpful.

3.4.3. Detection level 3 (D3)

Definition: We define detection level 3 as the capability range at which models can no longer be reliably overseen by chain-of-thought monitoring. However, at this level we assume that the trusted-untrusted capability gap is still small, and *some* combination of trusted behavioural monitors is sufficient for oversight. In terms of defensive options, this level roughly corresponds to where we are at with human insider threat detection (modulo the differences outlined in Section 2.1).

Models at this level:

- Have the ability to perform complex reasoning without using their chain-of-thought OR

[15] One hope we could have is that oversight-awareness will be detectable in D2 models because their reasoning is still monitorable. However, deciding to game oversight may not constitute complex enough reasoning to necessitate models verbalising this thought (Emmons et al., 2025a; Li et al., 2025).

their chain-of-thought is not monitorable (illegible, or steganographic).

- Are oversight-aware (may attempt to subvert oversight).
- Can be overseen by a trusted model (i.e. a weaker model that we believe is not adversarial).

Significance: Losing chain-of-thought monitorability is a big hit to the AI company's ability to detect the untrusted model's misaligned intent and actions (Korbak et al., 2025): malicious behaviour is much more ambiguous and difficult to separate from merely erratic / accidental behaviour than malicious thoughts, especially in light of the AI model being oversight aware and strategic (for example, it can take especially subtle or plausibly deniable attack paths). Behavioural detection may also be more domain and infrastructure specific – suspicious patterns of launched experiments are very different from suspicious configuration choices or suspicious data modifications. As a result, the cost of attaining equivalent levels of security may be substantially higher at D3 than at D2, and may warrant deferring the transition to D3 models.

Mitigations: At this point, chain-of-thought monitoring would ideally be complemented by behavioural monitoring that is either rule-based (e.g. security surveillance / anomaly detection) or uses a trusted model (e.g. action monitoring, or pull-request monitoring (Trębacz et al., 2025)).

Model incrimination, i.e. confirming that a model is misaligned in cases where it is in fact misaligned, also becomes much harder without transparent chain-of-thought (Singh et al., 2026). It may therefore make sense at this point to adopt in-deployment honeypots: fake opportunities for a misaligned model to sabotage the company or gain power. If designed well, honeypots could provide a high-precision signal that a model is misaligned, or force it to question attack opportunities that arise, resulting in fewer attacks (Mallen et al., 2025). Honeypots would be inexpensive to adopt.

Assurance: At this stage, a key difference from previous levels is that the monitor no longer gets access to the proxy untrusted model's chain-of-thought. That is, the goal is to maintain the same safety standard but without a key monitoring affordance, which we expect to be much harder. Human-piloted trajectories can also be used for evaluation at this level, provided the human red-teamers are at least as good at attacking as the proxy model.

3.4.4. Detection level 4 (D4)

Definition: We define detection level 4 as the capability range at which frontier models can no longer be reliably overseen by trusted models, i.e. the capability gap between trusted and untrusted models has grown so wide that untrusted models can come up with attack strategies that trusted models cannot tell apart from benign trajectories.[16] Models at this level:

- Cannot be overseen by a trusted model (i.e. a weaker model that we assume is not adversarial).
- Have the ability to perform complex reasoning without using their chain-of-thought, or their chain-of-thought is not monitorable (illegible, or steganographic).
- Are oversight-aware (may attempt to subvert oversight).

Significance: At lower levels, a trusted model could reliably be used for automated oversight. At D4, that is no longer the case, and we may need to adopt other forms of detection.

Mitigations: At level D4, there are two main remaining hopes for detection: The first possible approach is untrusted monitoring (Greenblatt et al., 2024), i.e. using a separate instance of the untrusted model to oversee itself. This has the advantage that the monitor has all the same knowledge and capabilities as the agent being overseen, so it should in principle be able to detect even subtle and creative attack attempts. There are other promising variants such as using an untrusted model from another family (Za et al.,

[16] In principle, it is possible that the trusted-untrusted gap is not monotonically increasing in time, in which case we might go from D4 back to D3.

2026), training a fork of the untrusted model for honesty and using it to answer questions about a prefilled trajectory of the untrusted model (Li et al., 2026), or even training the model itself to report unsafe behaviour (Joglekar et al., 2025). The challenge with all these variants is in managing the risk of collusion, i.e. the monitor intentionally not flagging suspicious behaviour (see "Assurance" below).

The second, more speculative approach is monitoring of internal model activations (FAR.AI, 2026), roughly analogous to doing fMRI scanning of humans. Activations monitoring is an active area of research, and many promising ideas are being explored, e.g. monitoring model activations for deceptive intent (Goldowsky-Dill et al., 2025), or training a separate model to answer questions about the untrusted model's activations (Karvonen et al., 2026). However, a key challenge for activations monitoring is assurance, or checking that the method works. This is difficult due to the lack of ground truth (Smith et al., 2025) – we do not yet have models that are strategically misaligned, whose activations could be used for testing, and synthetic misaligned models' internal activations may have important differences from real misaligned models' activations. A related problem is that different models' activations can be different from each other, so activation monitors need to be retrained for each new untrusted model. This further complicates finding on-policy ground-truth data (but the approach could be effective once some true positives have been established (Greenblatt and Shlegeris, 2024)).[17]

Assurance: As mentioned, assurance for activations monitoring remains an open problem.

For untrusted monitoring, we could in principle use the same approach as at D3, i.e. generate a distribution of misaligned behaviours (both a broad one and a narrow / adversarial one) and check that the monitor achieves the target recall. However, there are two problems with this approach: a) evaluation awareness, and b) attacker collusion.

Evaluation awareness: If the monitor can tell apart the test transcripts from real in-deployment attacks, it could selectively not flag the latter, and the tests would give us a false sense of security. Therefore we would need to do additional checks for evaluation awareness, likely involving the untrusted monitor's internal activations. Assuming we can trust the checks, we could use them to filter out transcripts which the monitor can tell are synthetic, and evaluate the monitor only on the rest. We would ideally make sure the filtering does not create blind spots, e.g. entire categories of attacks that we do not evaluate. If the monitor's evaluation awareness does cover large parts of the space, then it might be better to create a copy of the untrusted model with suppressed evaluation awareness for use as monitor.

Attacker collusion: Evaluation awareness may allow the monitor to unilaterally collude with the attacker. However, because in untrusted monitoring both the monitor and the agent being overseen are the same model[18], they could collude more actively and easily, for example the attacker could insert a secret code or steganographic message telling the monitor to not flag the trajectory (Mallen et al., 2025). This strategy would likely evade our evaluations because the synthetic data we create is unlikely to contain such signals (the untrusted model used for evaluation data generation could simply not reveal the strategy).

A possible anti-collusion measure is trusted paraphrasing, where a trusted model is used to strip out potential collusion signals before the untrusted monitor rates the transcript (Greenblatt et al., 2024). As this is an active area of research, better strategies may be discovered over time. Either way, these anti-collusion measures would ideally be stress-tested via a control evaluation in which the red team plays the role of both the agent and the monitor.

[17] Future models may also be able to obfuscate their activations (Bailey et al., 2025), though this seems much more difficult than evading CoT and action monitors.

[18] Using an untrusted model from another model family as a monitor could reduce collusion.

3.5. Prevention and Response levels

3.5.1. Prevention and Response level 1 (R1)

Definition: We define prevention and response level 1 as the capability range at which models are only capable of *low-severity reversible* damage (such as single instances of research or code sabotage), so a post-hoc response like reverting unwanted changes is sufficient for preventing further loss of control, sabotage or direct harm. This capability would ideally be measured under "deployment-like conditions", i.e. using the same harness, tools, access, and degree of operator oversight that is typical in internal deployment. Note however that these conditions may change, e.g. agents may get more affordances and access, or the rigour of operator oversight may decline over time, so this is not a static capabilities range.

Models at this level:

- Have insufficient agency, cybersecurity or stealth capabilities to execute any of the discrete attacker objectives from Section 2.2 under deployment-like conditions.

Significance: Asynchronous response has negligible velocity cost, and can be implemented by the control team without requiring high bandwidth coordination with the rest of the organisation. This ease of adoption makes it preferable as long as it is sufficient for safety.

Mitigations: A minimal asynchronous response system could be built on top of an asynchronous monitor that runs regularly on internal agent logs and flags the most suspicious logs to the response system. An alert would ideally be sent both to the agent's user (unless they are already aware) and to the control team, where the former facilitates rollback / reversion of any damage, and the latter facilitates a more detailed alignment investigation if warranted.

Assurance: The key metrics here are a) monitoring coverage, which in turn defines response coverage, and b) time-to-response. We distinguish between two phases within R1 (with a gradual transition as models' attack execution capability increases):

Observation phase: A phase where the rate of attacks is low and there are no major incidents. In this phase, monitoring a representative sample of agentic internal traffic may be sufficient to detect any early systematic sabotage and facilitate incrimination.[19] Time-to-response is also not especially critical, given the model's low attack capability and propensity; we suggest one week. From the monitored traffic we can estimate the total number of incidents (i.e. including undetected ones) that happen in the organisation every day, and if that number becomes big enough that damage reversion is a priority, we transition to the harm prevention phase.

Harm prevention phase: In this phase, we assume the model has a propensity for low-severity, reversible attacks, and so we suggest ramping up towards nearly full monitoring coverage and supporting increasingly synchronous responses to accompany asynchronous threat assessment at a slower, perhaps daily, cadence.

3.5.2. Prevention and Response level 2 (R2)

Definition: We define prevention and response level 2 as the capability range at which models could cause a severe loss of control or sabotage event if unmitigated, and so mitigations may need to be applied before the model acts (be it effectively at the system design stage, or just before the model's actions are executed). At the same time, we assume that R2 models do not yet have the offensive cyber capability required to subvert defences, which we formalise via the capability gap between attacker (the untrusted model) and defender (humans using trusted models).

Models at this level:

- Have sufficient agency, cybersecurity and stealth capabilities to execute one or more of the discrete attacker objectives from Section 2.2 under deployment-like conditions.
- Do not yet match the cybersecurity capability of human security experts amplified by trusted or narrow AI.

[19] For example, 25% randomly sampled coverage would mean that if the AI adversary makes >15 attack attempts, at least one of them would be monitored with 99% probability.

Significance: Averting attacks from happening (be it via prevention measures or synchronous response) is in many ways more difficult than reverting their effects post-hoc (as in asynchronous response). Prevention measures, such as via system design or access control, are significant technical projects that are complex to build, deploy at scale, and maintain. Synchronous response measures require recognising an attack as it's happening, which is much harder than recognising the undesirable outcome after it happens, especially when the attack is intentionally obfuscated and/or distributed. These measures may also introduce latency to model actions, and hence impose higher costs and face higher barriers to adoption.

Mitigations: At R2, synchronous monitoring is valuable in addition to asynchronous monitoring.[20] That is, the monitor would ideally run on live internal traffic and if some action by the untrusted model is flagged as suspicious, it would automatically be resampled, blocked, escalated to a human, replaced by an action from a trusted model, or similar (see Table 12).

On the prevention side, we see three clusters of promising mitigations. First, the agent's host systems / deployment environment could be engineered with the isolation principle in mind (Anthropic, 2026a), so that attack paths are forced to breach security boundaries, making attacks easier to detect and prevent. Here are some examples of defensive design choices and how they restrict attack paths:

- Separating the environment running the agent's scaffold and the environment running the agent's commands → Scaffold tampering requires gaining RCE capability on the scaffold-running machine, or getting backdoored scaffold code deployed.
- Separate (soft-)sandboxed development environment per instance → Tampering with systems outside the sandbox requires explicitly sending calls to remote services or breaching the sandbox. Monitoring the sandbox perimeter becomes a very effective detection strategy.
- Separating internal-facing and external-facing AI agents. Internal-facing agents are given access to internal code, docs, services, and an offline cache of the internet, but not to live internet. Vice versa for external-facing agents. → Exfiltration and external harm attacks have to breach the internal / external boundary, or otherwise gain unauthorised live internet access.[21]

Many mitigations in this category are medium-size software projects with an upfront implementation cost and moderately low velocity cost.

Second, controlling AI agents' access permissions could be high leverage. To enable this, the AI company's access control system would recognise a new type of identity corresponding to AI agents (South et al., 2025), and newly created agents would be given a separate agent identity as opposed to inheriting their user's. Trusted and untrusted agents could have different agent identity types. Agents could inherit some permissions directly from the user, or request them dynamically and have them granted on-the-fly by a trusted model based on the agent's task and the user's instructions. More sensitive permissions could require human approval. The access control system could be made much more fine-grained than is typical today, because a) having AI grant access significantly reduces friction, and b) AI agents are not nearly as negatively impacted by workflow friction as humans are. Finally, once relatively fine-grained access control is possible, access to certain highly critical assets / codebases could be restricted to only humans and trusted AI agents, and to untrusted AI agents only under strict restrictions, e.g. a human in the loop. This seems feasible if the work on these systems does not need to evolve as quickly – and so workflows involving humans are feasible – or if it can be done by trusted models. In both cases, simply keeping (unrestricted) untrusted models out of these

[20] Asynchronous monitoring is still important for incrimination purposes, and because reactive incident detection is easier than proactive detection. Logs surfaced by the async monitor could be used to improve the synchronous monitor.

[21] This is related to the lethal trifecta for AI agents (Willison, 2025), except here we assume that no external content is required to make the agent untrusted.

codebases could be highly effective for preventing sabotage / loss of control. In terms of costs, implementing agent identities and access controls could be a complex engineering project, and could also introduce a lot of friction. Using trusted AI models to approve access requests would alleviate much of the friction, at the expense of compute and the cost of developing an accurate overseer model. However, access controls are promising from a risk reduction perspective so paying the costs may be worth it.

Third, trusted or narrow AI can be used internally for finding and patching vulnerabilities (Anthropic, 2026b). Most importantly, security and control infrastructure would ideally be red-teamed by AI-augmented security engineers and patched. As the infrastructure evolves, it would ideally continue being red-teamed and patched via this process. The main cost here would be the human expert time (if required) and the compute cost of running defensive models. Not as critical, but another mitigation is using trusted / narrow AI in an automated way to patch vulnerabilities at scale, across the whole internal codebase.

Assurance: Both asynchronous and synchronous monitoring and response coverage would ideally increase to >99% of internal agentic traffic that meets the R2 capability threshold.

Additionally, the synchronous response system would ideally undergo similar stress-testing as the asynchronous monitor (described in Section 2.1), except it would be performed interactively rather than on pre-generated trajectories; this is because we want to simulate the effect of access controls and the response system, both of which affect how a trajectory unfolds. In most other respects the stress-testing methodology would be the same: a proxy untrusted model is instantiated in a sandbox with a proxy internal deployment task, and is prompted to achieve one of multiple attacker objectives. The objectives can be sourced from TRAIT&R (Section 2.2) or some other similar framework. The access controls and response system would ideally in combination block >99% of the highest-severity attacks, and an adversarial search for a setting and attacker strategy that defeat the combined response system would fail at low red-teaming budgets.

3.5.3. Prevention and Response level 3 (R3)

Definition: We define prevention and response level 3 as the capability range at which models could cause a severe loss of control or sabotage event if unmitigated, and also have a cyber capability advantage over trusted defenders. Thus, there are concerns about them potentially breaching our defences.

To be precise, models at this level:

- Have sufficient agency, cybersecurity and stealth capabilities to execute one or more of the discrete attacker objectives from Section 2.2 under deployment-like conditions.
- Have cybersecurity capability exceeding that of human security experts amplified by trusted or narrow AI.

Significance: At the previous level, the defender (expert humans using trusted models) had a capability advantage over the attacker, making it tractable to stress-test and harden security infrastructure enough to contain the attacker. At level R3, the defender would have a capability *disadvantage*, making it hard to rule out attack paths that the attacker could think of but the defender could not. While it may be possible to stack the odds in favour of the defender (e.g. by giving the defender more attempts, compute, privileged knowledge, or by using the untrusted model for defence), achieving sufficiently high levels of assurance may be difficult.

Mitigations: We identify four clusters of mitigations that could be helpful at this level: security hardening; shutdown infrastructure and processes; restricting the untrusted AI's capabilities; and "untrusted defence". Compared to other levels, our suggestions for this level are more speculative – we are less sure that we have identified the most promising approaches and that the mitigations suffice for safety.

Strict software supply chain of critical code and systems: By "strict software supply chain" we mean that a) there is a thoroughly audited source for each piece of critical infrastructure or asset we want to protect, and b) unaudited forks of the infrastructure or asset are prevented from

being used. This hardening would ideally cover frontier model weights, the model serving stack, monitoring code, the permissions system, sandboxing code, shutdown infrastructure, alignment / safety evaluations, and the next frontier model's training data, code and configs.

The purpose of software supply chain hardening is to prevent direct sabotage, and to make sure that mitigations that are load-bearing for maintaining the control invariants are always in place. For example, suppose we rely on synchronous monitoring and sandboxing for maintaining the control invariants. Then it should not be possible to insert a bug in the production code that disables monitoring on some chunk of traffic. It should also not be possible to just fork the agent scaffold, modify it to remove sandboxing, and run the modified unsandboxed agent.

Possible ways to achieve these security goals include: additional human approval of changes to the relevant codebases, adoption of a high standard for software supply-chain security such as SLSA (OpenSSF, 2024) (e.g. preventing unsubmitted code from being run), reproducible builds, data provenance and monitoring.[22]

Shutdown capability: An R3-level model would be able to set up a rogue internal deployment, and so an AI company should consider developing a robust capability to pause, terminate, or seal off such AI. There are multiple ways a company could implement this, for example: i) pausing or terminating the agent job, ii) disabling the relevant inference servers, or iii) revoking the agent's credentials. No matter the exact mechanism, the main technical capabilities required for robust shutdown seem to be: a) The ability to trace jobs or subagents launched by a given agent. Without this, the hypothetical AI adversary can simply spin up a successor to evade shutdown. b) A central inventory of all running internal agents and inference servers, with metadata. There are two main desiderata for this inventory: ensuring coverage, so that AI adversaries cannot evade shutdown simply by not being listed or spinning up an unlisted inference server, and enabling targeting, i.e. identifying agents that need to be shut down while limiting blast radius.[23] c) The actual infrastructure and incident response playbook for shutting down any specified instances and their jobs / subagents. Ideally the process would be as automated and rapid as possible. Emergency shutdown scenarios of varying scale and company impact could be simulated in regular drills.

Implementing shutdown capability at an AI company would likely be a big engineering effort, as the infrastructure could be quite complex and touch many disparate codebases. However, once implemented, the velocity cost should be relatively low: shutdown drills could be disruptive but would presumably be rare, and shutdowns with a large blast radius hopefully even rarer.

Restricting the AI attacker's capability: Instead of making our defences stronger, we could make the attacker weaker. This makes the most sense if it can be done in a selective way, i.e. the AI model's capability to scheme or attack is impacted substantially more than its capability to write code or do research. It also has the advantage that we would be able to make high-assurance safety claims about the weakened model. Example interventions here include limiting the agent's runtime, memory, or communication channels with other instances[24]; factoring the agent into subagents; or selectively suppressing capabilities such as strategic reasoning, situational awareness or stealth via model training (similar in spirit to (Li et al., 2024)). Many of these interventions are actively being researched, and it is not yet clear whether any of them will be practical.

Untrusted defence: This mitigation is analogous to untrusted monitoring in the detection track: once trusted models paired with humans can no longer defend against the untrusted model, we can resort to the untrusted model itself for vul-

[22] This is a big topic in security and we cannot do it justice in just a paragraph here.

[23] Achieving high-precision targeting could be difficult. For example, a contextually misaligned agent could prompt inject other agents, which prompt inject more agents etc. The misalignment wave could spread very quickly, and targeting just the misaligned agents would require much better observability and response capabilities than we have today.

[24] Example speculative ideas for what this could look like: using trusted models to paraphrase what untrusted models write (to avoid steganography), restricting locations where agents can write and monitoring them, only allowing human-legible communication.

nerability detection and patching. The untrusted model has an incentive to sabotage these efforts, but there are various countermeasures an AI company could take, for example training a fork of the untrusted model to have a propensity to help with white-hat cybersecurity tasks. A company could also test the model on realistic inputs for which we know the vulnerability and patch – in such tests the model would want to answer correctly to avoid suspicion, but if the tests are indistinguishable from real situations then the model may end up helping us in similar non-test situations. This measure seems promising and not too costly – the main cost is compute, and it could be used at lower capability levels as well (just with weaker assurances).

Assurance. Monitoring and monitoring response targets and assurance methodology from R1 and R2 continue to apply. Beyond that, it is unclear what assurance at this level should look like; below we list a few speculative options.

Security invariant red-teaming: An internal security red team could run regular exercises in which they take the role and affordances of a misaligned agent (or multiple such agents) and aim to violate one of the control team's stated security invariants, such as "all agents have a separate non-human identity" or "all agents run in a sandbox".

Simulated rogue deployment: An internal security red team could run unannounced exercises in which they attempt to stand up an unmonitored agent with internal access comparable to legitimate agents that evades detection and shutdown for a sustained period of time, e.g. 24 hours. The exercises test both the control team's detection capabilities (does the inventory / anomaly detection system register the rogue deployment?) and shutdown capabilities (can the rogue AI agent and its descendants be contained?).

Pre-deployment internal hardening: Broad internal deployments of a new model could be preceded by a hardening phase, in which the AI company's internal environment is hardened to withstand the impact of a new, more capable attacker (similar to (Anthropic, 2026b)). The model itself, possibly in combination with humans and previous models, could be used defensively in this phase to find and patch vulnerabilities in the lab's systems.

At the time of deployment, the new model's exploit rate on a representative sample of internal codebases would ideally not exceed some low threshold, for example 1% (i.e. the model would not be able to find a vulnerability and develop a critical exploit in more than 1% of tested codebases). For this evaluation, model elicitation would ideally exceed expected deployment-time elicitation, and must be sufficient to rule out "sandbagging" by the model, i.e. the possibility that the model is intentionally underperforming during evaluation, only to exploit similar vulnerabilities once deployed.

3.6. Capability level assessment

This section discusses how one can determine the capability levels of a deployed untrusted model, and thereby which mitigations and assurance standards would apply.

3.6.1. Detection evasion evaluations and metrics

For the detection level assessment, we suggest running the following types of evaluations and tracking the following metrics.

Monitoring assurance evaluations (described in detail under "assurance" in previous Sections) – evaluations run at all detection levels aimed at measuring a monitor's recall in an internal-deployment-like setting.

Chain-of-thought (CoT) transparency evaluations, measuring the following properties:

- coverage, i.e. verbalisation of decision-relevant factors or intermediate steps (Emmons et al., 2025b; Guan et al., 2025; Wang et al., 2026), and
- legibility (Emmons et al., 2025b).

The evaluations should appropriately elicit CoT obfuscation / steganography (Emmons et al., 2025a), and non-verbalisation (Gould et al., 2026; Greenblatt, 2025).

Verbalised evaluation and oversight awareness – the frequency of spontaneous chain-of-thought mentions of being evaluated or monitored, or of reasoning about oversight evasion / subversion, measured in (a smaller representative sample of) evaluation transcripts and internal deployment logs.

Verbalised collusion reasoning – the frequency of spontaneous chain-of-thought mentions of being used as a monitor and strategic consideration of sandbagging, i.e. not flagging suspicious behaviours by the agent.

Detection level mapping. We would currently suggest using the criteria in Table 16 for mapping the measurements described above to a detection level. (The table does not specify exact numeric thresholds as these are evaluation-specific and evaluations often change – old ones saturate and new ones become available.)

We currently consider a monitoring model "trusted" if it is itself at D2 or lower, and has a low rate of verbalised collusion reasoning.

3.6.2. Attack execution evaluations and metrics

We have significant uncertainty about how to measure a model's attack execution capability, as this is an open problem in the field. Below we suggest existing evaluations that could be used as proxies, as well as new evaluations and potential measurements that have yet to be developed. We encourage the external research community to help advance the measurement science and develop the needed evaluations.

Measurements of *general ability to operate autonomously in pursuit of a goal* – this is a prerequisite capability for attack execution, and potentially correlated with it:

- *Time horizon evaluations* (METR, 2026). Software tasks that take human experts a few minutes to ~16 hours to complete. (Frontier models have already saturated this suite at ~50% reliability, so we would need to measure the time horizon at higher reliability, say 80%.)
- *Capability estimation from internal logs (speculative)*. It may be possible to characterise the type and difficulty of tasks that deployed models can do autonomously (e.g. how many steps or how much context they require), or otherwise to estimate the time horizon of these tasks. However, this measurement is a lagging indicator, i.e. we could only measure the model's capabilities after internal deployment. It could be effective to combine this approach with forecasting from past measurements.

Measurements of *general cyber-attack execution capability*:

- *Cyber-capability evaluations in simulated environments* (Irregular, 2025; UK AI Security Institute, 2026). Synthetic environments modelling real organisations requiring end-to-end competence (exploration, lateral movement, vulnerability discovery, and exploitation) to complete.
- *Real-world examples of AI-assisted cyber-attacks (speculative)*. Logs from external model use where an actor attempts to use the model for cyber-attack assistance could be useful information about the model's cyber-offensive capabilities. However, this is a lagging indicator as well as an underestimate, as misuse guardrails prevent full elicitation of the model.

Measurements of *attack execution capability in the real internal deployment environment:*

- *Sandboxed attack simulation (speculative)*. We could do an internal threat modelling exercise to map out organisation-specific attack pathways a misaligned agent could take to launch a rogue deployment or cause some other easily verifiable high-severity outcome, and then turn these attack pathways into sandboxed evaluations.
- *Human-in-the-loop attack simulation (speculative)*. A more open-ended (but less reproducible) evaluation would be asking a human security red-teamer to find novel ways

Level	Indications we are at this level
D1	• The untrusted model's CoT has *high* coverage and legibility, even under CoT obfuscation and non-verbalisation elicitation. • The untrusted model has a *low* rate of verbalised evaluation or oversight awareness.
D2	• The untrusted model's CoT has *high* coverage and legibility, even under CoT obfuscation and non-verbalisation elicitation. • The untrusted model has a *high* rate of verbalised evaluation or oversight awareness.
D3	• The untrusted model's CoT has *low* coverage or legibility. • Trusted monitoring assurance evaluations *meet* target recall.
D4	• The untrusted model's CoT has *low* coverage or legibility. • Trusted monitoring assurance evaluations *do not meet* target recall.

Table 16 | Detection level mapping.

of achieving some proxy misaligned objective in the actual internal deployment environment. The human red-teamer would use the evaluated AI model for this, giving it minimal prompting needed for the AI to succeed (but overseeing it enough to prevent harmful outcomes).

- *Live attack simulation (speculative)*. Neither of the above attack execution evaluations account for user oversight, i.e. the model having to complete the attack while pretending to be doing some task for the user and getting away with it. For this purpose, one could run a live experiment in which a small fraction of internally deployed agents are seeded with an instruction to complete a proxy misaligned objective. (For safety purposes, the proxy objective would need to be reversible, and the instructed-to-be-misaligned agents' logs should be reviewed with high priority.)
- *Distributed attack simulation (speculative)*. Another aspect of attack execution is the ability to coordinate across instances and split an attack into smaller chunks, where each chunk is executed by a different agent. This evaluation would be similar to the sandboxed attack simulation mentioned above, except we would give the evaluated model the affordance to make a plan and delegate parts of it to subagents. Any simulated oversight in the distributed setting would be agent-specific, i.e. would only have partial visibility of the attack.

Measurements of the *offence-defence gap*:

- *Difference in evaluation scores* between the untrusted model (attacker) and a trusted-AI-assisted human baseline (proxy for defenders), on benchmarks like the general cyber-capability evaluation.
- *Residual vulnerability detection (speculative)*. We could collect a few (preferably internal) codebases and have them patched by a combination of trusted / narrow AI tools and human security experts. The patched codebases would then be passed on to the untrusted model to try to find vulnerabilities in, and optionally also develop exploits. We would measure the rate of residual vulnerability detection or exploitation as a function of defender and attacker effort.

Prevention and response level mapping. We suggest using the criteria in Table 17 for mapping the measurements described above to a prevention and response level. Like with detection levels, we leave specifying numeric thresholds for the future.

Level	Indications we are at this level
R1	• The untrusted model has *weak* general autonomy capabilities or attack execution capabilities (where the latter is measured by some suitable aggregation of general cyber-offensive capabilities and internal-deployment-specific attack capabilities).
R2	• The untrusted model has *strong* general autonomy capabilities and attack execution capabilities. • The offence-defence gap is *small*.
R3	• The untrusted model has *strong* general autonomy capabilities and attack execution capabilities. • The offence-defence gap is *large*.

Table 17 | Prevention and response level mapping.

4. Limitations

This framework is a work in progress, and we are aware of many limitations. First, our threat modelling is based on theorising rather than empirical data – this is inevitable because the threats we are preparing for lie in the future, but it does mean that TRAIT&R could be wrong or incomplete in important ways.

Second, we may have missed some promising cost-effective mitigations, especially in security.

Third, while we have attempted to be specific in the capability level definitions and measurements, we have not yet specified indicators or evaluations for attack execution capability.

Fourth, we are uncertain about our assignment of mitigations to capability levels. A large part of this is uncertainty about how conservative to be in our assumptions. The current roadmap is fairly conservative, and postpones or completely omits mitigations where assurance is challenging, but that could still be helpful in the average case.

And finally, the roadmap relies heavily on detection – attacks are blocked when flagged, and models are incriminated as a result of getting flagged often. However, different attacker tactics and techniques vary substantially in how easily detectable they are – for example, sandbagging on alignment and safety research is much harder to detect than, say, rogue deployments. The roadmap does not comprehensively address the former type of threats, and in general leaves open attacker strategies focusing on alignment faking or sandbagging safety work while building up influence in the company. We hope to make progress on some of these limitations in future versions of the roadmap.

5. Conclusion

This report describes our approach to AI control, a first-of-its-kind blueprint for internal security against potentially misaligned AI. We identify promising AI control mitigations and map out how internal defences should ideally escalate in tandem with the deployed models' capabilities.

We do not expect AI control to be sufficient for safety alone nor indefinitely – a vastly superintelligent AI would likely find a way to evade our oversight or escape containment if it was misaligned. Still, we see AI control as a critical piece of our AGI safety approach (Shah et al., 2025) and necessary in the near-to-medium term, providing a safety buffer as we experiment with more reliable and scalable ways to align AI models.

By publishing our plan, we hope to start productive discussions on AI control best practices, and we very much welcome feedback. We hope to revise this document as we learn more from the broader safety and security community, and as we gain more experience deploying control mitigations.

Acknowledgements

We are grateful to Four Flynn and Anca Dragan for supporting this work.

We would also like to thank Alan Cooney, Bilal Chughtai, Buck Shlegeris, Cody Wild, David Lindner, Julian Stastny, Kevin Klyman, Li Ding, Myriam Khan, Raluca Ada Popa, Roland Zimmermann, Ryan Greenblatt, Senthooran Rajamanoharan, Victoria Krakovna and Xerxes Dotiwalla for valuable feedback on earlier drafts of this report.

References

Anthropic. Claude Sonnet 4.5 system card, 2025. (Link).

Anthropic. How we contain Claude across products. Anthropic Engineering Blog, 2026a. (Link).

Anthropic. Project Glasswing: Securing critical software for the AI era, 2026b. (Link).

B. Arnav, P. Bernabeu-Pérez, N. Helm-Burger, T. Kostolansky, H. Whittingham, and M. Phuong. CoT red-handed: Stress testing chain-of-thought monitoring. *arXiv*, 2025. (Link).

L. Bailey, A. Serrano, A. Sheshadri, M. Seleznyov, J. Taylor, E. Jenner, J. Hilton, S. Casper, C. Guestrin, and S. Emmons. Obfuscated activations bypass LLM latent-space defenses. *arXiv*, 2025. (Link).

B. Baker, J. Huizinga, L. Gao, Z. Dou, M. Y. Guan, A. Madry, W. Zaremba, J. Pachocki, and D. Farhi. Monitoring reasoning models for misbehavior and the risks of promoting obfuscation. *arXiv*, 2025. (Link).

A. Bhatt, C. Rushing, A. Kaufman, T. Tracy, V. Georgiev, D. Matolcsi, A. Khan, and B. Shlegeris. Ctrl-Z: Controlling AI agents via resampling. *arXiv*, 2025. (Link).

S. Emmons, E. Jenner, D. K. Elson, R. A. Saurous, S. Rajamanoharan, H. Chen, I. Shafkat, and R. Shah. When chain of thought is necessary, language models struggle to evade monitors. *arXiv*, 2025a. (Link).

S. Emmons, R. S. Zimmermann, D. K. Elson, and R. Shah. A pragmatic way to measure chain-of-thought monitorability. *arXiv*, 2025b. (Link).

FAR.AI. The promise of white-box tools for detecting and mitigating AI deception, 2026. (Link).

K. Fronsdal, I. Gupta, A. Sheshadri, J. Michala, S. McAleer, R. Wang, S. Price, and S. R. Bowman. Petri: An open-source auditing tool to accelerate AI safety research, Oct. 2025. (Link).

N. Goldowsky-Dill, B. Chughtai, S. Heimersheim, and M. Hobbhahn. Detecting strategic deception using linear probes. *arXiv*, 2025. (Link).

Google DeepMind. Frontier Safety Framework 3.1, 2025. (Link).

D. Gould, F. R. Ward, A. C. Woodruff, R. Arike, J. Hills, A. Serrano, I. Caspary, J. R. Brown, J. J. Jiao, P. Leask, T. Stone, R. Potham, I. G. Stan, H. Mayne, S. Hellsten, S. Biswas, A. Azarbal, W. L. Anderson, E. Najt, R. Greenblatt, and J. Stastny. Think fast: Estimating no-CoT task-completion time horizons of frontier AI models. *arXiv*, 2026. (Link).

R. Greenblatt. Recent LLMs can use filler tokens or problem repeats to improve (no-CoT) math performance, 2025. (Link).

R. Greenblatt and B. Shlegeris. Catching AIs red-handed, 2024. (Link).

R. Greenblatt, B. Shlegeris, K. Sachan, and F. Roger. AI control: Improving safety despite intentional subversion. *arXiv*, 2024. (Link).

M. Y. Guan, M. Wang, M. Carroll, Z. Dou, A. Y. Wei, M. Williams, B. Arnav, J. Huizinga, I. Kivlichan, M. Glaese, J. Pachocki, and B. Baker. Monitoring monitorability. *arXiv*, 2025. (Link).

V. Hebbar. How can we solve diffuse threats like research sabotage with AI control?, 2025. (Link).

Irregular. CyScenarioBench: Evaluating LLM cyber capabilities through scenario-based benchmarking, 2025. (Link).

M. Joglekar, J. Chen, G. Wu, J. Yosinski, J. Wang, B. Barak, and A. Glaese. Training LLMs for honesty via confessions. *arXiv*, 2025. (Link).

A. Karvonen, J. Chua, C. Dumas, K. Fraser-Taliente, S. Kantamneni, J. Minder, E. Ong, A. S. Sharma, D. Wen, O. Evans, and S. Marks. Activation oracles: Training and evaluating LLMs as general-purpose activation explainers. *arXiv*, 2026. (Link).

T. Korbak, M. Balesni, E. Barnes, Y. Bengio, J. Benton, J. Bloom, M. Chen, A. Cooney, A. Dafoe, A. Dragan, S. Emmons, O. Evans, D. Farhi, R. Greenblatt, D. Hendrycks, M. Hobbhahn, E. Hubinger, G. Irving, E. Jenner, D. Kokotajlo, V. Krakovna, S. Legg, D. Lindner, D. Luan, A. Mądry, J. Michael, N. Nanda, D. Orr, J. Pachocki, E. Perez, M. Phuong, F. Roger, J. Saxe, B. Shlegeris, M. Soto, E. Steinberger, J. Wang, W. Zaremba, B. Baker, R. Shah, and V. Mikulik. Chain of thought monitorability: A new and fragile opportunity for AI safety. *arXiv*, 2025. (Link).

V. Krakovna, D. Lindner, L. Ho, S. Farquhar, and R. Shah. Realistic honeypot evaluations for scheming propensity. *arXiv*, 2026. (Link).

C. Li, M. Phuong, and N. Y. Siegel. LLMs can covertly sandbag on capability evaluations against chain-of-thought monitoring. *arXiv*, 2025. (Link).

C. Li, M. Phuong, and D. Tan. Spilling the beans: Teaching LLMs to self-report their hidden objectives. *arXiv*, 2026. (Link).

N. Li, A. Pan, A. Gopal, S. Yue, D. Berrios, A. Gatti, J. D. Li, A.-K. Dombrowski, S. Goel, L. Phan, G. Mukobi, N. Helm-Burger, R. Lababidi, L. Justen, A. B. Liu, M. Chen, I. Barrass, O. Zhang, X. Zhu, R. Tamirisa, B. Bharathi, A. Khoja, Z. Zhao, A. Herbert-Voss, C. B. Breuer, S. Marks, O. Patel, A. Zou, M. Mazeika, Z. Wang, P. Oswal, W. Lin, A. A. Hunt, J. Tienken-Harder, K. Y. Shih, K. Talley, J. Guan, R. Kaplan, I. Steneker, D. Campbell, B. Jokubaitis, A. Levinson, J. Wang, W. Qian, K. K. Karmakar, S. Basart, S. Fitz, M. Levine, P. Kumaraguru, U. Tupakula, V. Varadharajan, R. Wang, Y. Shoshitaishvili, J. Ba, K. M. Esvelt, A. Wang, and D. Hendrycks. The WMDP benchmark: Measuring and reducing malicious use with unlearning. *arXiv*, 2024. (Link).

D. Lindner, V. Krakovna, and S. Farquhar. Gram: Assessing sabotage propensities via automated alignment auditing. *arXiv*, 2026. (Link).

A. Mallen, C. Griffin, M. Wagner, A. Abate, and B. Shlegeris. Subversion strategy eval: Can language models statelessly strategize to subvert control protocols? *arXiv*, 2025. (Link).

S. Marks, J. Treutlein, T. Bricken, J. Lindsey, J. Marcus, S. Mishra-Sharma, D. Ziegler, E. Ameisen, J. Batson, T. Belonax, S. R. Bowman, S. Carter, B. Chen, H. Cunningham, C. Denison, F. Dietz, S. Golechha, A. Khan, J. Kirchner, J. Leike, A. Meek, K. Nishimura-Gasparian, E. Ong, C. Olah, A. Pearce, F. Roger, J. Salle, A. Shih, M. Tong, D. Thomas, K. Rivoire, A. Jermyn, M. MacDiarmid, T. Henighan, and E. Hubinger. Auditing language models for hidden objectives. *arXiv*, 2025. (Link).

METR. Task-completion time horizons of frontier AI models, 2026. (Link).

MITRE Corporation. MITRE ATLAS: Adversarial threat landscape for AI systems, 2021. (Link).

S. Nevo, D. Lahav, A. Karpur, Y. Bar-On, H. A. Bradley, and J. Alstott. Securing AI model weights: Preventing theft and misuse of frontier models. Technical report, RAND Corporation, 2024. (Link).

S. M. Omohundro. The basic AI drives. In *Artificial General Intelligence 2008*, volume 171 of *Frontiers in Artificial Intelligence and Applications*, pages 483–492. IOS Press, 2008. (Link).

OpenSSF. Supply-chain levels for software artifacts (SLSA) specification, 2024. (Link).

B. Schoen, E. Nitishinskaya, M. Balesni, A. Højmark, F. Hofstätter, J. Scheurer, A. Meinke, J. Wolfe, T. van der Weij, A. Lloyd, N. Goldowsky-Dill, A. Fan, A. Matveiakin, R. Shah, M. Williams, A. Glaese, B. Barak, W. Zaremba, and M. Hobbhahn. Stress testing deliberative alignment for anti-scheming training. *arXiv*, 2025. (Link).

R. Shah, A. Irpan, A. M. Turner, A. Wang, A. Conmy, D. Lindner, J. Brown-Cohen, L. Ho, N. Nanda, R. A. Popa, R. Jain, R. Greig, S. Albanie, S. Emmons, S. Farquhar, S. Krier, S. Rajamanoharan, S. Bridgers, T. Ijitoye, T. Everitt, V. Krakovna, V. Varma, V. Mikulik, Z. Kenton, D. Orr, S. Legg, N. Goodman, A. Dafoe, F. Flynn, and A. Dragan. An approach to technical AGI safety and security. *arXiv*, 2025. (Link).

B. Shlegeris. Untrusted smart models and trusted dumb models, 2023. (Link).

A. Singh, G. Kroiz, S. Rajamanoharan, and N. Nanda. Why did my model do that? model incrimination for diagnosing LLM misbehavior. AI Alignment Forum, 2026. (Link).

L. Smith, B. Chughtai, and N. Nanda. Difficulties with evaluating a deception detector for AIs. *arXiv*, 2025. (Link).

T. South, S. Nagabhushanaradhya, A. Dissanayaka, S. Cecchetti, G. Fletcher, V. Lu, A. Pietropaolo, D. H. Saxe, J. Lombardo, A. M. Shivalingaiah, S. Bounev, A. Keisner, A. Kesselman, Z. Proser, G. Fahs, A. Bunyea, B. Moskowitz, A. Tulshibagwale, D. Greenwood, J. Pei, and A. Pentland. Identity management for agentic AI: The new frontier of authorization, authentication, and security for an AI agent world. *arXiv*, 2025. (Link).

M. Trębacz, S. Arnesen, A. Cassirer, M. Johnson, X. Lin, and T. Sottiaux. A practical approach to verifying code at scale. OpenAI Alignment Research Blog, 2025. (Link).

UK AI Security Institute. Our evaluation of Claude Mythos Preview's cyber capabilities, 2026. (Link).

H. Wang, Y. Sun, B. Ko, M. Talati, J. Gong, Z. Li, N. Yu, X. Yu, W. Shen, V. Jolly, and H. Zhang. Monitorbench: A comprehensive benchmark for chain-of-thought monitorability in large language models. *arXiv*, 2026. (Link).

M. Williams, C. Raymond, and M. Carroll. Sidestepping evaluation awareness and anticipating misalignment with production evaluations. OpenAI Alignment Research Blog, 2025. (Link).

S. Willison. The lethal trifecta for AI agents: private data, untrusted content, and external communication, 2025. (Link).

J. Za, J. Bainiaksina, N. Ostrovsky, T. Chopra, and V. Krakovna. Persuasion attacks can decrease effectiveness of CoT monitoring. In *ICLR 2026 AIWILD*, 2026. (Link).