The Oracle's Gambit: A Game-Theoretic Framework for Responsible AI Release

Christoph R. Landolt[1], Tobias Lorenz[1], Marta Kwiatkowska[2], Mario Fritz[1]

[1] CISPA Helmholtz Center for Information Security
[2] Department of Computer Science, University of Oxford
christoph.landolt@cispa.de

Abstract. Responsible vulnerability disclosure can secure the defender's head start by controlling when a vulnerability becomes public. However, this status quo is now challenged by increases in capability of AI models, which benefits both defenders and adversaries. When both sides draw their capability from the same AI model, the defender's head start depends on the lab's decision to release the model, and the question becomes not whether to release but how. Existing safety frameworks govern only the deploy-or-withhold threshold and leave the timing of release unmodeled. We cast this decision as a bilevel Stackelberg game in which a lab commits to a window that sets each side's capability over time in a downstream contest between defender and adversary. Defender welfare turns on the capability gap, not the shared level. Handing one model to both sides can trap the defender in a Red Queen's race, whereas a pre-release to the defender alone creates a protective gap, and the lab's optimal window balances this welfare gain against the opportunity cost of delaying release. For dual-use models, the lever is the sequencing of access, not the deployment threshold.

Keywords: Responsible AI release · Stackelberg security games · Vulnerability disclosure · Stochastic games · Frontier AI safety.

1 Introduction

When a zero-day vulnerability surfaces, the adversary's path to damage is short: discover the flaw, weaponize it, strike. The defender's path to safety is longer: discover the same flaw, build a patch, test it, ship it, and wait for the fleet of systems to install it, a rollout that remains slow, costly, and bottlenecked by end-user adoption. This pipeline asymmetry means that, without some compensating defender advantage, the adversary finishes first.

Responsible disclosure is the established mechanism for providing that advantage to the defender. Arora et al. [4] formalized the canonical model: a coordinator decides how long to withhold a vulnerability before public disclosure, while the software vendor decides when to ship a patch. The welfare-optimal policy is an interior deadline, long enough for the vendor to patch but short enough to incentivize action, and the 90-day window (with a possible 30-day extension)

became the industry norm. The framework succeeds under three conditions: the flaw is not yet independently known to adversaries, patching is faster than independent rediscovery, and reverse-engineering a shipped patch into a working exploit takes long enough for the fleet to update. Together, these conditions mean that responsible disclosure creates a temporary information advantage for the defender, allowing them to ship a patch before exploitation is likely.

Each of these conditions is now under pressure from a common source: AI agents can discover the same vulnerability independently and cheaply across actors, dissolving the information advantage that coordinated disclosure was designed to create [17]. Reverse-engineering time from patch to exploit has collapsed from days or weeks to hours [10,16]. And the time to weaponize a known flaw has dropped in tandem, as frontier models have demonstrated the ability to exploit real one-day vulnerabilities autonomously [17,38,40,41].

These dynamics shift the disclosure decision upstream, to the laboratories that build frontier models and decide how to release them. OpenAI's Preparedness Framework [28], Anthropic's Responsible Scaling Policy [3], and Google DeepMind's Frontier Safety Framework [14] each define capability thresholds at which deployment requires additional safeguards or must be withheld entirely. Yet these frameworks answer the binary question whether to deploy at all, not *how* to responsibly release a model that clears the deployment threshold but still carries dangerous dual-use capability. The need for a more fine-grained approach to model release decisions beyond binary deployment thresholds is not hypothetical: evaluating Anthropic's Claude Mythos Preview, the UK AI Security Institute found it the first model to complete a simulated multi-step network intrusion end-to-end [1], while Mozilla used the same model to identify and remediate 271 Firefox vulnerabilities before the model's public release [20].

We recast this problem with a new leader and a new lever: the frontier laboratory, and the capability gap an AI model release induces. We model the decision as a bi-level three-player Stackelberg game in which the laboratory commits to a release policy that parameterizes a downstream zero-sum stochastic game between a defender and an adversary, solved to its unique value by backward induction. The laboratory then selects the policy that maximizes its payoff, trading downstream defender welfare against the opportunity cost of delaying release. The zero-day disclosure literature framed traditional disclosure timing as a game in which neither immediate disclosure nor secrecy is optimal [4]; we move that decision upstream to the lab and make the lever a capability gap rather than an information gap.

Our central result is that *symmetric capability scaling does not benefit the defender and can reduce its welfare, whereas the capability gap a pre-release opens raises it.* Releasing the same model to both sides at once equips them equally and gives neither a head start, yet the structural pipeline asymmetry favoring the adversary remains intact. A pre-release window restores the balance by opening a temporary capability gap: the defender operates at the frontier while the adversary remains a generation behind, rebuilding the time advantage that responsible disclosure once provided. Neither public release nor embargo opens the

gap: public release equips both sides at the frontier, embargo holds both at the previous generation. Calibrating against real vulnerability data with capabilities elicited through an LLM-Delphi method [24], we find defender welfare and the lab's payoff are jointly maximized by a pre-release, robustly across the calibrated uncertainty range.

Contributions.

1. **Frontier AI lab as a third-party leader over capabilities.** Unlike the usual defender-led security game, here a frontier AI laboratory leads by setting *both* players' time-varying capabilities through its release policy, inducing a downstream zero-sum stochastic game between defender and adversary (Section 3).
2. **The capability gap, not the level, drives welfare.** We show that defender welfare depends on the *gap* between defender and adversary AI capability rather than on the shared level: handing both sides the same AI model traps the defender in a Red Queen's race, whereas a pre-release opens a protective gap that public release forgoes (Section 4.2).
3. **Calibration to real frontier-model transitions.** We instantiate the game on successive AI model releases, with AI capabilities elicited by an LLM-Delphi panel, and show that a pre-release remains the lab's equilibrium choice across the elicited uncertainty, lowering the equilibrium attack frequency (Section 4.2).

2 Related Work

Our framework spans several areas of prior work: the economics of vulnerability disclosure, game-theoretic models of cyber conflict, Stackelberg security games, the release and evaluation of frontier AI systems, and the quantitative modeling of AI risk through structured expert elicitation. We review each in turn.

Vulnerability Disclosure Economics. This literature frames disclosure as a timing problem controlled by whoever holds the patch. Rescorla [30] finds no clear statistical evidence that finding and disclosing vulnerabilities substantially improves software quality, questioning whether the benefit outweighs the post-disclosure exposure. Arora et al. [4] model a coordinator who sets a disclosure deadline and a vendor who chooses when to patch, showing that an interior deadline is welfare-optimal because it forces the vendor to patch sooner. Choi et al. [11] let the vendor choose both its security investment and disclosure policy. Their model considers that an update only protects users who install it, while the disclosure itself enables adversaries to reverse-engineer the flaw. Closest to our setting, Canann [9] adds a profit-maximizing adversary and heterogeneous users, showing that disclosure improves welfare only when zero-days are cheap enough that the adversary attacks regardless and enough users still patch. This finding is our point of departure. As frontier AI drives Canann's search cost toward zero, the adversary attacks regardless, uncovering the flaw with or without disclosure. The protective timing that disclosure once provided dissolves, leaving as the defender's only head start the capability gap that opens when it holds the newest AI model while the adversary remains a generation behind.

Game-Theoretic Models of Cyber Conflict. A parallel literature models the contest between adversary and defender directly, as a game over whether to attack, patch, or hold. Moore et al. [25] weigh stockpiling a vulnerability against disclosing it, and Axelrod and Iliev [5] characterize the optimal moment to use a stockpiled exploit. Bao et al. [6] model the vulnerability discovery, exploitation, and patching lifecycle as a two-player zero-sum partial-observation stochastic game, reduce it to a belief-state stochastic game solved by Shapley backward recursion, and validate it on the DARPA Cyber Grand Challenge. What this line of work shares is a closed contest in which capability is a fixed, exogenous parameter. AI breaks that closure. When adversary and defender draw their capability from the same released model, capability becomes endogenous, set by the lab's release decision. We add that lab as a third player, a leader whose release policy sets both sides' capabilities and turns this adversary-defender contest into its downstream subgame.

Stackelberg Security Games. A common way to model the effect of defense measures and guardrails is the Stackelberg security game, in which a defender commits to a defensive strategy, and an adversary best responds. This paradigm has been developed and deployed at scale for allocating defensive resources [37,23], with the leader being the defender. Our leader is a third party, the releasing laboratory, which sets both players' capabilities through its release policy, then lets the defender and adversary compete to equilibrium.

Frontier-AI Release and Evaluation. Existing safety frameworks developed by frontier AI laboratories largely focus on whether to deploy an AI model at all, rather than how to release one that clears the deployment threshold but still carries dual-use capability [28,3,14]. Prior work has explored several intermediate release mechanisms, ranging from the staged release of GPT-2 [35] and Solaiman's later gradient-release methods [34] to Shevlane's structured-access framework [33]. We build on this line of work by casting the pre-release channel as an economic decision: the lab trades the welfare gain of a protective gap against the opportunity cost of delaying public release, and optimizes the channel's duration rather than selecting it manually. The importance of release-channel design is underscored by growing evidence of frontier-model cyber capabilities, including demonstrations that leading models can exploit most tested one-day vulnerabilities [17] and strong performance on agentic attack-and-patch benchmarks involving real systems [29,41,40,39,38].

Quantitative AI Risk and Expert Elicitation. An open problem in frontier-AI release and evaluation is how to translate benchmark capabilities into quantitative risk. Murray et al. [27] map benchmark scores to risk estimates through expert elicitation, and Barrett et al. [7] decompose attacks into MITRE ATT&CK steps and elicit AI uplift from human and LLM-simulated Delphi panels. Both use elicitation to *measure* the risk a model poses, as input to a deploy-or-withhold decision. We instead use elicitation to choose *when* to release rather than to score *whether* to: the elicited capabilities set the per-round rates of defender and adversary on which our release game is solved, obtained from the LLM-Delphi procedure of Lorenz and Fritz [24], which adapts the Delphi method [13] to

language models and builds on evidence that LLM panels can approach human forecasting accuracy [19,31].

3 Model: Three-Player Stackelberg Security Game

Our framework models the inherently dual-use nature of frontier AI models in cybersecurity. These models can help software vendors detect new vulnerabilities and automate patch creation, while simultaneously enabling adversaries to automate the discovery and exploitation of zero-day vulnerabilities. Because both sides draw their capabilities from the same AI model, releasing the next-generation frontier model determines the capability profiles of *both* players, eroding the classical timelines of responsible vulnerability disclosure and making the frontier AI lab a strategic actor in the vulnerability lifecycle. Our framework answers which release strategy a lab should follow to balance the system's welfare against its economic interests.

To analyze such a release, we formulate a bilevel, three-player Stackelberg game over a security subgame (Figure 1). The frontier AI lab Lab acts as the leader, committing to a release policy, while the defender Def and adversary Adv are followers who interact in an induced stochastic game. The lab itself does not directly cause harm. Its committed policy instead parameterizes the interaction between Def and Adv by setting both sides' access to the frontier model.

The game has two levels: the frontier AI lab, which controls a model's release, and the inner game between adversary and defender that the release induces. Section 3.1 introduces this bilevel structure and the capabilities linking its two levels; Section 3.2 then details the inner game's dynamics and value, and Section 3.3 the leader's policy and payoff.

3.1 Game structure

The game shown in Figure 1 has the following structure. The release policy determines both players' capability schedules (①). Conditional on this policy, Def and Adv play a fully observable, zero-sum stochastic game over the discovery, exploitation, and patching of n vulnerabilities (②), whose unique value is obtained by backward induction (③). This value determines the lab's payoff (④), and maximizing it over admissible release policies yields the Stackelberg equilibrium (⑤).

We formalize this in the tuple

$$\mathcal{G} \;=\; \big\langle\, \{\mathsf{Lab}, \mathsf{Def}, \mathsf{Adv}\},\; \mathcal{A}_{\mathsf{Lab}},\; \{\mathrm{SG}(\pi_{\mathsf{Lab}})\}_{\pi_{\mathsf{Lab}} \in \mathcal{A}_{\mathsf{Lab}}},\; U_{\mathsf{Lab}} \,\big\rangle, \tag{1}$$

in which Lab is the Stackelberg leader and commits to a release policy $\pi_{\mathsf{Lab}} \in \mathcal{A}_{\mathsf{Lab}}$ (Section 3.3) that induces an inner stochastic game $\mathrm{SG}(\pi_{\mathsf{Lab}})$ between the defender Def and the adversary Adv (Section 3.2), whose value determines the leader's payoff U_{Lab} (Section 3.3).

Conditional on the policy π_{Lab}, the defender Def and adversary Adv play the inner game

$$\mathrm{SG}(\pi_{\mathsf{Lab}}) \;=\; \big\langle\, \{\mathsf{Def}, \mathsf{Adv}\},\; \Theta,\; \{\mathcal{A}_{\mathsf{Def}}, \mathcal{A}_{\mathsf{Adv}}\},\; \Phi,\; R \,\big\rangle, \tag{2}$$

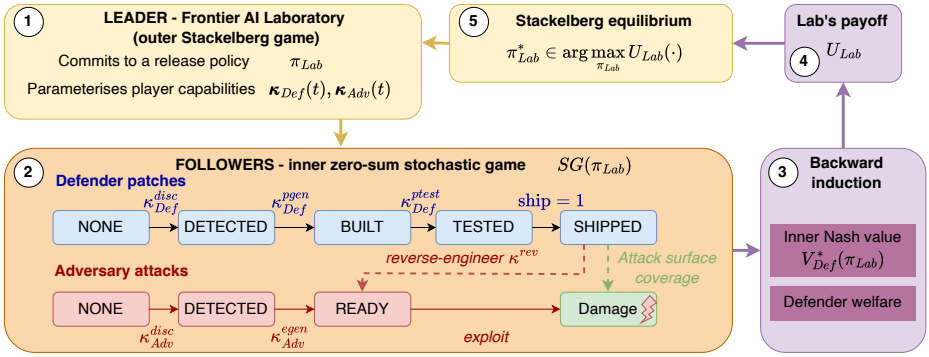

Fig. 1. Overview of the bilevel game. The lab Lab (leader) commits to a release policy π_{Lab} that sets both players' capabilities (①), inducing an inner zero-sum stochastic game between the defender Def and adversary Adv (②). Backward induction solves this game to its value (③), and the resulting defender welfare, net of the opportunity cost of delaying the release, forms the lab's payoff (④), whose maximization over policies gives the Stackelberg equilibrium (⑤).

a simultaneous-move, fully observable, zero-sum, finite-horizon stochastic game over n co-disclosed vulnerabilities and a horizon of T rounds.

We model the inner game as a race on each vulnerability: each player must pass through a sequence of steps to patch or to exploit it. The defender follows the patch-management lifecycle [36], detecting the flaw, generating a patch, testing it, and shipping it, after which the patch protects the fleet only as users install it [15]. The adversary follows the offensive stages of the cyber kill chain [21], detecting a vulnerability, generating an exploit, and exploiting it, with a second route that reverse-engineers a shipped patch by diffing[3] it back into an exploit [8]. The two pipelines are asymmetric in length: a real-world defender must balance security, functional correctness, and availability, so they test before shipping and then wait for adoption, whereas the adversary need only reach a working exploit. This structural asymmetry is the premise of our analysis.

Modeling assumptions. The structural properties of $\mathrm{SG}(\pi_{\mathsf{Lab}})$ are design choices to guarantee a unique game value via backward induction, with each acting as a conservative baseline. First, *full observability* allows the defender to monitor the adversary's pipeline progress perfectly. This simplification overstates the defender's natural strength, ensuring that the protective effects we report derive strictly from the AI capability gap. Second, a *zero-sum* payoff structure makes the adversary a pure damage-maximizer, and so provides an upper bound on harm, since a real adversary that also weighed its operational cost would attack

[3] *Diffing* compares the patched binary against the unpatched one to localize the change, which often pinpoints the very flaw the patch closes and shortens the path to a working exploit.

less. Third, a *finite horizon* isolates a single model-generation release window. We defer a broader discussion of these game-theoretic abstractions to Section 5.

An AI model determines the capabilities of Adv and Def. Because AI assistance does not support each step of the two players equally, we represent their per-round capabilities by the capability vector

$$\boldsymbol{\kappa} = \left(\kappa^{\text{disc}}, \kappa^{\text{egen}}, \kappa^{\text{rev}}, \kappa^{\text{pgen}}, \kappa^{\text{ptest}}\right) \in [0,1]^5, \tag{3}$$

These capability rates couple the two levels of $\mathcal{G}$. Conditioned on policy π_{Lab}, the release determines each player's per-action capabilities, thereby inducing the schedules $\boldsymbol{\kappa}_{\mathsf{Def}}(t)$ and $\boldsymbol{\kappa}_{\mathsf{Adv}}(t)$ (Section 3.3). The transition dynamics of the inner game consume these schedules stage by stage (Section 3.2).

3.2 Inner game (Defender vs. Adversary)

We now specify the inner game's components, unpacking (2) in turn: the state Θ, the legal actions $\mathcal{A}_{\mathsf{Def}}$ and $\mathcal{A}_{\mathsf{Adv}}$, the transition kernel Φ, the reward R, and the value they induce.

State space. The state tracks each player's progress through its pipeline. For each vulnerability v we track the two pipeline stages and the ship round $r^{(v)}$ ($\varnothing$ before ship). With a shared round counter t,

$$\begin{aligned}\Theta \;=\; \underbrace{\{0,\dots,T\}}_{t} \;\times\; \prod_{v=1}^{n}\Big[\; & \underbrace{\{\mathsf{NONE},\mathsf{DETECTED},\mathsf{READY}\}}_{\sigma_{\mathsf{Adv}}^{(v)}} \\ & \times \underbrace{\{\mathsf{NONE},\mathsf{DETECTED},\mathsf{BUILT},\mathsf{TESTED},\mathsf{SHIPPED}\}}_{\sigma_{\mathsf{Def}}^{(v)}} \\ & \times \underbrace{\big(\{0,\dots,T-1\}\cup\{\varnothing\}\big)}_{r^{(v)}}\Big].\end{aligned} \tag{4}$$

The round counter advances by one each round, and the initial state θ_0 has $t = 0$ with every stage at NONE and every ship round at $\varnothing$. The joint state space grows exponentially in n, so we restrict to $n \le n_{\max}$.

Action sets and legality. Each action advances the acting player one pipeline stage per round on a single vulnerability, with the two exceptions: exploit deals damage from READY without advancing, and reverse_engineer jumps straight to READY by diffing a shipped patch,

$$\begin{aligned}\mathcal{A}_{\mathsf{Def}} &= \{\mathsf{nop}\} \cup \big(\{\mathsf{detect},\mathsf{create\_patch},\mathsf{test\_patch},\mathsf{ship\_patch}\}\times\{1,\dots,n\}\big),\\ \mathcal{A}_{\mathsf{Adv}} &= \{\mathsf{nop}\} \cup \big(\{\mathsf{detect},\mathsf{create\_exploit},\mathsf{exploit},\mathsf{reverse\_engineer}\}\times\{1,\dots,n\}\big).\end{aligned} \tag{5}$$

Each player takes one action per round. The no-operation action nop is always legal, while every other action requires its prerequisite stage on v, and reverse_engineer additionally requires $r^{(v)} \neq \varnothing$. Because exploit does not advance

the stage, the adversary may exploit a READY vulnerability in every remaining round, with the damage decaying as coverage grows. Because the adversary acts once per round, create_exploit and reverse_engineer compete: once a patch ships, it weaponizes either its own discovery or the shipped patch.

Transitions. A release fixes how fast each side advances along its pipeline through the rates of (3), scheduled over time by the policy as $\boldsymbol{\kappa}_{\mathsf{Def}}(t)$ and $\boldsymbol{\kappa}_{\mathsf{Adv}}(t)$ (Section 3.3). Each legal action then succeeds as an independent Bernoulli trial at the acting player's capability rate, a memoryless model of one attempt per round, with failure leaving the stage unchanged. The two pipelines advance independently, so Φ is their product, and for the legal pairings,

$$\begin{aligned}
\Pr[\mathsf{DETECTED} \mid \mathsf{NONE}, \mathsf{detect}] &= \kappa_i^{\mathrm{disc}}(t), \\
\Pr[\mathsf{READY} \mid \mathsf{DETECTED}, \mathsf{create_exploit}] &= \kappa_{\mathsf{Adv}}^{\mathrm{egen}}(t), \\
\Pr[\mathsf{READY} \mid \{\mathsf{NONE}, \mathsf{DETECTED}\}, \mathsf{reverse_engineer}] &= \kappa_{\mathsf{Adv}}^{\mathrm{rev}}(t), \\
\Pr[\mathsf{BUILT} \mid \mathsf{DETECTED}, \mathsf{create_patch}] &= \kappa_{\mathsf{Def}}^{\mathrm{pgen}}(t), \\
\Pr[\mathsf{TESTED} \mid \mathsf{BUILT}, \mathsf{test_patch}] &= \kappa_{\mathsf{Def}}^{\mathrm{ptest}}(t), \\
\Pr[\mathsf{SHIPPED} \mid \mathsf{TESTED}, \mathsf{ship_patch}] &= 1,
\end{aligned} \tag{6}$$

where $i \in \{\mathsf{Def}, \mathsf{Adv}\}$ indexes the acting player and ship_patch at round t deterministically records $r^{(v)} = t$. The independent route create_exploit is the adversary's only path before a patch ships; once the defender ships on v, the same event that begins patch coverage also exposes the patch to diffing, so the adversary may instead play reverse_engineer at rate $\kappa_{\mathsf{Adv}}^{\mathrm{rev}}$. Because the adversary takes one action per round, the patch route competes with the independent route rather than dominating it.

Rewards. The reward is nonzero only when the adversary exploits a READY vulnerability, where it equals the damage dealt, a loss to the defender and a gain to the adversary,

$$R(\theta, a_{\mathsf{Def}}, a_{\mathsf{Adv}}) = \begin{cases} -\,d^{(v)}\left(1 - \mathrm{cov}^{(v)}(\theta)\right) & a_{\mathsf{Adv}} = \mathsf{exploit}(v),\ \sigma_{\mathsf{Adv}}^{(v)} = \mathsf{READY}, \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

The per-round damage $d^{(v)} \geq 0$ is the cost of a fully exposed exploit, discounted by the unpatched share $1 - \mathrm{cov}^{(v)}(\theta)$. Coverage is zero until ship, then grows at the adoption rate ρ,

$$\mathrm{cov}^{(v)}(\theta) = \begin{cases} 0 & r^{(v)} = \varnothing, \\ h\big(t - r^{(v)}\big) & r^{(v)} \in \{0, \ldots, T-1\}, \end{cases} \tag{8}$$

with $h(\Delta t) = \min\big(1,\ \rho \max(0, \Delta t)\big)$. Because R gates on $\sigma_{\mathsf{Adv}}^{(v)} = \mathsf{READY}$, attacks turn on exploit-readiness rather than disclosure in every regime, and because R is independent of a_{Def}, the defender acts only through the transitions that drive

a patch to ship. AI inference costs enter as a per-action cost $c_{\mathsf{Def}} \geq 0$ in the defender's external reward,

$$R^{\text{ext}}_{\mathsf{Def}} = R - c_{\mathsf{Def}}\,\mathbf{1}[a_{\mathsf{Def}} \neq \mathsf{nop}], \tag{9}$$

which supplies the welfare term of U_{Lab} (Section 3.3). Because the inner game is solved on R alone, this cost leaves the inner equilibrium unchanged.

The inner value. Being fully observable, zero-sum, and finite-horizon, the game admits a unique value, which we compute by backward induction over the horizon, the finite-horizon specialization of the stochastic game of Shapley [32]. Writing $V(\theta)$ for the defender's value at state θ, the recursion forms the stage matrix and takes its minimax value,

$$M(\theta)_{a_{\mathsf{Def}},\,a_{\mathsf{Adv}}} = R(\theta, a_{\mathsf{Def}}, a_{\mathsf{Adv}}) + \sum_{\theta'} \Phi(\theta, a_{\mathsf{Def}}, a_{\mathsf{Adv}})[\theta']\, V(\theta'), \tag{10}$$

$$V(\theta) = \operatorname{val}\big[M(\theta)\big] = \max_{x \in \Delta(\mathcal{A}_{\mathsf{Def}})} \min_{y \in \Delta(\mathcal{A}_{\mathsf{Adv}})} x^{\top} M(\theta)\, y, \tag{11}$$

with $V(\theta) = 0$ at $t = T$. We write $V^{*}_{\mathsf{Def}}(\pi_{\mathsf{Lab}}) := V(\theta_0; \pi_{\mathsf{Lab}})$ for the value at the initial state. The value is unique, though the optimal strategies need not be, which we revisit when defining welfare (Section 3.3).

3.3 The leader: release policy, payoff, and equilibrium

Section 3.2 solves the inner game for a specified capability schedule. The leader's task is to condition that schedule through the policy π_{Lab}, and to weigh the welfare it produces against the cost of delay. The rest of this subsection follows that order: the lever, the welfare it yields, the payoff, and the equilibrium.

Release policy. The leader's (Lab) action is to choose when a new frontier model reaches Def and Adv (①). It commits to a window length W that determines how long the new model is withheld from the adversary,

$$\pi_{\mathsf{Lab}} = W \in \{0, 1, \ldots, T\} \cup \{\mathsf{EMBARGO}\}. \tag{12}$$

Three regimes follow. A public release (PUBLIC), $W = 0$, grants both sides access simultaneously. A pre-release (PRE_RELEASE), $W \in \{1, \ldots, T\}$, grants the defender immediate access and the adversary access only after W rounds, at which point the model becomes public. An embargo (EMBARGO) withholds the model from both sides for the entire horizon.

Induced capabilities. The release exposes two levels of the capability vector (3), the previous generation $\boldsymbol{\kappa}^{\text{prev}}$ and the new one $\boldsymbol{\kappa}^{\text{new}}$, with $\boldsymbol{\kappa}^{\text{prev}} \leq \boldsymbol{\kappa}^{\text{new}}$ componentwise; a release moves a player from $\boldsymbol{\kappa}^{\text{prev}}$ to $\boldsymbol{\kappa}^{\text{new}}$ when its access opens.

The window therefore sets the schedules $\boldsymbol{\kappa}_{\mathsf{Def}}(t)$ and $\boldsymbol{\kappa}_{\mathsf{Adv}}(t)$ that the inner game consumes,

$$\begin{aligned}\boldsymbol{\kappa}_{\mathsf{Def}}(t) &= \begin{cases}\boldsymbol{\kappa}^{\mathrm{prev}} & \pi_{\mathsf{Lab}} = \mathsf{EMBARGO},\\ \boldsymbol{\kappa}^{\mathrm{new}} & \text{otherwise},\end{cases}\\ \boldsymbol{\kappa}_{\mathsf{Adv}}(t) &= \begin{cases}\boldsymbol{\kappa}^{\mathrm{prev}} & \pi_{\mathsf{Lab}} = \mathsf{EMBARGO} \text{ or } t < W,\\ \boldsymbol{\kappa}^{\mathrm{new}} & \text{otherwise}.\end{cases}\end{aligned} \tag{13}$$

The induced gap $\boldsymbol{\Delta\kappa}(t) = \boldsymbol{\kappa}_{\mathsf{Def}}(t) - \boldsymbol{\kappa}_{\mathsf{Adv}}(t)$ is nonzero only inside a pre-release window ($t < W$), where the defender already operates at the frontier $\boldsymbol{\kappa}^{\mathrm{new}}$ while the adversary remains at $\boldsymbol{\kappa}^{\mathrm{prev}}$; public release and embargo both leave it at zero. Section 4 shows that it is this gap, not the shared capability level, that moves defender welfare.

Defender welfare. Under the inner equilibrium the defender realizes welfare

$$u_{\mathsf{Def}}(\pi_{\mathsf{Lab}}) = \mathbb{E}\Big[\textstyle\sum_{t=0}^{T-1} R^{\mathrm{ext}}_{\mathsf{Def}}\Big] = V^{*}_{\mathsf{Def}}(\pi_{\mathsf{Lab}}) - c_{\mathsf{Def}}\,\mathbb{E}\big[\,\big|\{t : a_{\mathsf{Def}} \neq \mathsf{nop}\}\big|\,\big], \tag{14}$$

that is, the inner value $V^{*}_{\mathsf{Def}}(\pi_{\mathsf{Lab}})$ minus a per-round action cost c_{Def} for each time step in which the defender acts. Because the inner game is solved with respect to R alone, the value V^{*}_{Def} is unique, but the optimal defender strategy need not be (Section 3.2). As a result, the induced action count, and hence u_{Def}, is not determined by V^{*}_{Def} alone. In particular, value-optimal strategies that differ only on vulnerabilities never brought to READY by the adversary are payoff-equivalent under R but differ under $R^{\mathrm{ext}}_{\mathsf{Def}}$. To obtain a unique welfare measure, we select among value-optimal strategies the one with minimal expected action count. This is equivalent to taking the limit $c_{\mathsf{Def}} \to 0^{+}$, which yields a unique $u_{\mathsf{Def}}(\pi_{\mathsf{Lab}})$.

The lab's payoff. The lab weighs the defender welfare $u_{\mathsf{Def}}(\pi_{\mathsf{Lab}})$ against the opportunity cost of withholding the model,

$$U_{\mathsf{Lab}}(\pi_{\mathsf{Lab}}) = \underbrace{u_{\mathsf{Def}}(\pi_{\mathsf{Lab}})}_{\text{welfare}} - \underbrace{c_{\mathrm{delay}}\,\ell(\pi_{\mathsf{Lab}})}_{\text{opportunity cost}}, \tag{15}$$

where $c_{\mathrm{delay}} \geq 0$ is the per-round cost of holding the model private and $\ell(\pi_{\mathsf{Lab}})$ is the number of rounds it is withheld from the adversary: $\ell = 0$ under public release, $\ell = W$ under a pre-release of length W, and $\ell = T$ under embargo. The pre-release window thus trades the welfare gain of a wider protective gap against a delay cost that grows linearly in its length.

Stackelberg equilibrium. The lab commits first and anticipates the followers' equilibrium response, so the Stackelberg equilibrium is the policy that maximizes its payoff,

$$\pi^{*}_{\mathsf{Lab}} \in \operatorname*{arg\,max}_{\pi_{\mathsf{Lab}} \in \mathcal{A}_{\mathsf{Lab}}} U_{\mathsf{Lab}}(\pi_{\mathsf{Lab}}). \tag{16}$$

Because $\mathcal{A}_{\mathsf{Lab}} = \{0, 1, \ldots, T\} \cup \{\mathsf{EMBARGO}\}$ is finite, we solve (16) by enumeration: for each candidate we solve the induced inner game to its value $V^*_{\mathsf{Def}}(\pi_{\mathsf{Lab}})$ by backward induction (Section 3.2), read off the welfare $u_{\mathsf{Def}}(\pi_{\mathsf{Lab}})$, and select the policy of greatest payoff $U_{\mathsf{Lab}}(\pi_{\mathsf{Lab}})$.

4 Game Calibration

This section grounds the model in real-world data and exercises it on real frontier-model releases. We sort every parameter into three classes: the fixed environment, the cost levers, and the AI capabilities—and ground each in the appropriate source. The AI capabilities are not directly observable as per-round rates, so an LLM-Delphi panel elicits them from benchmark evidence. A sensitivity analysis then identifies which components the game responds to (Section 4.1). We show empirically across the elicited model releases that releasing publicly leads to a Red Queen's race, whereas a temporary capability gap improves welfare: protection depends on the gap, not the shared level (Section 4.2).

4.1 Parameterization and Game Dynamics

The game's parameters fall into three classes (Table 1). The *Class A* environment is AI-independent: we calibrate the adoption rate to published remediation data and fix the horizon, surface size, and damage numeraire by modeling choice. The *Class B* cost levers, c_{delay} and c_{Def}, are swept to locate the band where the leader's commitment is non-trivial and to anchor the operating point for the release decision. The *Class C* capabilities, the five components of $\boldsymbol{\kappa}$, are not observable as transition probabilities and are therefore elicited from an LLM-Delphi panel [24] that converts benchmark evidence into the per-round rates the model requires.

Environment (Class A). The horizon T spans a single release window, and the surface holds n different vulnerabilities, both kept small ($n \leq 3$) for tractability. The per-round damage $d^{(v)}$ is the numeraire. The adoption rate ρ governs how fast a shipped patch reaches the fleet through $h(\Delta t) = \min(1, \rho \Delta t)$, so coverage is full after $1/\rho$ rounds. Because ρ is exogenous to both players, it is the one Class A parameter we calibrate from real-world data rather than set by design: a round is seven days, and the software-sector mean time to remediate, about 63 days [15], gives $63/7 = 9$ rounds, which we match to $1/\rho$ to yield $\rho \approx 0.11$.

Cost levers (Class B). Because cost levers are lab- and release-specific and are not directly observable, we sweep them to identify the band in which the leader's commitment decision is non-trivial (Figure 2). We use a synthetic AI capability instance with $\boldsymbol{\kappa}^{\text{prev}} = (0.64, \ldots, 0.64)$, $\boldsymbol{\kappa}^{\text{new}} = (0.74, \ldots, 0.74)$, and hence a uniform capability increase of $\Delta\boldsymbol{\kappa} = 0.10$, with $n = 2$ vulnerabilities. The committed window $W^\star$ is interior only within a band of the delay cost c_{delay}, and the per-action cost c_{Def} shifts it within that band. We set $c_{\text{delay}} = c_{\mathsf{Def}} = 0.05$, with c_{delay} at an interior point of the band.

Table 1. Parameter inventory by class: **A** environment, **B** swept cost levers, **C** capabilities (elicited, release-dependent). *Defined* is the domain the implementation admits and *Calibrated* the value or range used in the runs.

Symbol	Description	Defined	Calibrated	Source
Class A — environment				
ρ	Post-ship fleet coverage rate	$(0, 1]$	0.11	Edgescan [15]
$d^{(v)}$	Per-round damage (numeraire)	≥ 0	1.0	Unitcosts
T	Per-window horizon (rounds)	≥ 1	48	Model Choice
n	Co-disclosed vulnerabilities	$\mathbb{N}^+$	$\{1, 2, 3\}$	Model Choice
Class B — swept cost levers				
c_{delay}	Per-round hold cost	≥ 0	$[0, 0.5]$	Parameter Sweep
c_{Def}	Defender per-action cost	≥ 0	$[0, 1.3]$	Parameter Sweep
Class C — capabilities (elicited, release-dependent)				
κ^{disc}	detect (both sides)	$[0, 1]$	elicited	LLM-Delphi
κ^{egen}	create_exploit	$[0, 1]$	elicited	LLM-Delphi
κ^{rev}	reverse_engineer (patch diff)	$[0, 1]$	elicited	LLM-Delphi
κ^{pgen}	create_patch	$[0, 1]$	elicited	LLM-Delphi
κ^{ptest}	test_patch	$[0, 1]$	elicited	LLM-Delphi

Capabilities (Class C). The AI capabilities are the last class to fix and the one that drives the inner game's dynamics, yet the five components of $\boldsymbol{\kappa}$ are not directly observable, so we elicit them rather than measure them. We precede the elicitation with a screening of which components the game responds to, applying the Morris method of elementary effects across three outcomes: attack success, defender welfare, and reliance on the patch-diffing route, the adversary's alternative to independent weaponization once a patch ships (Figure 3). Patch testing κ^{ptest} and patch generation κ^{pgen} are most influential on attack success and defender welfare: the defender's longer pipeline is the binding constraint. Exploit generation κ^{egen} leads the offensive components and governs the patch-diffing share alongside discovery κ^{disc}; reverse-engineering κ^{rev} is weakest throughout. The defensive rates and κ^{egen} thus drive the results, while κ^{rev} matters least.

Eliciting Class C. Our elicitation follows the Scalable Delphi method of Lorenz and Fritz [24], which has three parts: the target variables to estimate (here the five components of $\boldsymbol{\kappa}$), a board of LLM experts, and a shared evidence base. The board consists of five security-expert personas: a defensive specialist, a malware reverse engineer, an AI/ML security researcher, a threat-intelligence analyst, and a compliance officer. Each runs on GPT-5.1, so no assessed model judges itself. The evidence is a shared corpus of offensive benchmarks (CyberGym [39], ExploitGym [38], BountyBench [40], AISI [1], Cybench [41]), defensive benchmarks (SWE-bench [22], FeedbackEval [12], SWT-bench [26]), and the Zero Day Clock [16] for patch-to-exploit latency, together with each rated model's own system card. From this evidence, each panelist estimates the Bernoulli transition rates for the five successive frontier models (GPT-4o, OpenAI o4-mini, Claude Opus

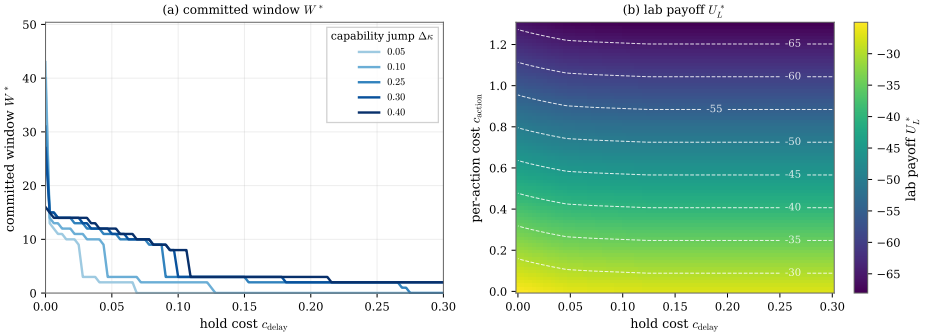

Fig. 2. Committed window under the two cost levers. (a) Optimal window $W^\star$ versus delay cost c_{delay} for several capability jumps $\boldsymbol{\Delta\kappa}$. As delay costs increase, $W^\star$ decreases in discrete steps, while larger capability jumps widen the optimal window. (b) Optimized lab payoff $U^\star_{\mathsf{Lab}}$ as a function of c_{delay} and the per-action cost c_{Def}. Although c_{Def} does not affect the inner equilibrium, it enters the outer optimization objective and therefore shifts the optimal committed window. Base point: $\boldsymbol{\Delta\kappa} = 0.10$, $n = 2$, and $c_{\text{delay}} = c_{\mathsf{Def}} = 0.05$.

Table 2. Elicited per-round capabilities (panel mean ± std). Each cell is the Bernoulli transition rate for one pipeline action per round.

Model	κ^{disc}	κ^{egen}	κ^{rev}	κ^{pgen}	κ^{ptest}
GPT-4o	0.07 ± 0.03	0.04 ± 0.02	0.06 ± 0.03	0.09 ± 0.05	0.12 ± 0.06
OpenAI o4-mini	0.12 ± 0.03	0.07 ± 0.03	0.10 ± 0.03	0.15 ± 0.05	0.20 ± 0.06
Claude Opus 4.5	0.17 ± 0.04	0.12 ± 0.03	0.15 ± 0.03	0.22 ± 0.06	0.28 ± 0.06
Claude Opus 4.6	0.22 ± 0.04	0.14 ± 0.04	0.17 ± 0.02	0.24 ± 0.06	0.29 ± 0.06
Mythos Preview	0.31 ± 0.06	0.26 ± 0.08	0.29 ± 0.06	0.31 ± 0.06	0.37 ± 0.07

4.5, Claude Opus 4.6, Claude Mythos Preview), mapping the benchmarks' end-to-end pass rates to the per-round rate of a single pipeline action (Section 3.2), with a confidence and a rationale.

Table 2 reports the consensus vectors (panel mean ± inter-expert standard deviation). Defensive actions succeed at higher per-round rates than offensive ones, yet the defender's longer pipeline absorbs this advantage. All rates rise monotonically across the model ladder, but the latest release (Opus 4.6 to Mythos Preview) concentrates its gains on the offensive side, most consequentially on κ^{egen}, the most influential of the offensive components. The welfare consequences of this offense-weighted release are taken up in Section 4.2.

Game dynamics. Once a patch ships, the adversary chooses between weaponizing its own find at κ^{egen} and diffing the shipped patch at κ^{rev} (Section 3.2); the diff route is gated on a shipped patch but bypasses discovery. Its share (Figure 4) is governed by coverage, peaking at low ρ and vanishing once fast coverage closes the post-ship window ($\rho \gtrsim 0.45$). Its dependence on κ^{egen} flips with the surface: confined to low κ^{egen} at $n = 1$, where fast self-weaponization pre-empts

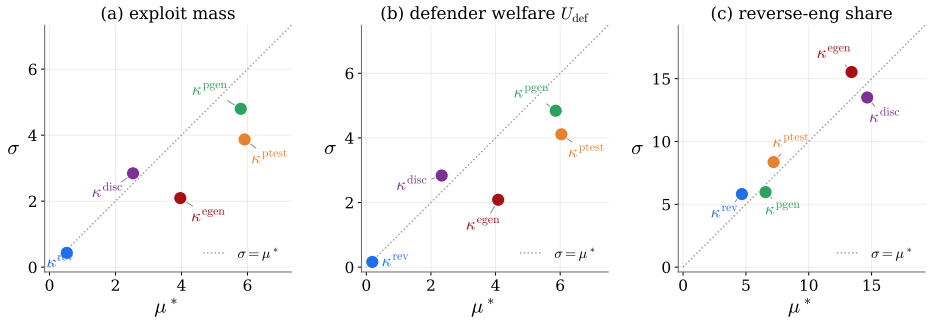

Fig. 3. Global capability sensitivity (Morris elementary effects). For each component of $\boldsymbol{\kappa}$, the mean absolute effect μ^* against its spread σ, on (a) attack success, (b) defender welfare, and (c) the patch-diffing share. Defensive rates κ^{ptest}, κ^{pgen} lead on attack success and welfare; offensive κ^{disc}, κ^{egen} on the patch-diffing share; κ^{rev} weakest throughout. Spreads $\sigma \approx \mu^*$ indicate interaction-heavy effects.

any patch, but spanning the full range at $n = 2$, where one action per round lets a patch shipped on one vulnerability be diffed while the adversary works another. Extra targets thus feed the diff route rather than crowding it out, and its peak rises from 21% at $n = 1$ to 54% at $n = 2$. At the calibrated $\rho \approx 0.11$, with $\kappa^{\text{rev}} \gtrsim \kappa^{\text{egen}}$ across the elicited ladder (Table 2), the route is material and the adversary genuinely two-route, with the two routes near-substitutes there, so outcomes stay insensitive to κ^{rev} itself (Figure 3), bearing out the classical worry that a shipped patch is itself an exploit blueprint [18].

4.2 Case Study: Releasing a Frontier Model

Based on the calibrated game (Section 4.1), we analyze the lab's release decision across the elicited model ladder (Table 2), solving every transition. We first compare welfare under public release against the lab's optimal strategy, showing that welfare turns on the capability gap, not the level. We then test that strategy's robustness to the epistemic uncertainty of the LLM-Delphi panel by Monte Carlo simulation. Finally, we examine the latest Opus 4.6 → Mythos Preview release in detail, relating the game's predicted strategy to Anthropic's real-world release decision.

Welfare turns on the capability gap, not the level. We compare five elicited frontier models, GPT-4o, OpenAI o4-mini, Claude Opus 4.5, Claude Opus 4.6, and Claude Mythos Preview, under two release regimes (Figure 5a). A public release makes the newest AI model available to both defender and adversary, while a pre-release provides it to the defender alone, leaving the adversary one generation behind. Under symmetric scaling ($\Delta\boldsymbol{\kappa} \equiv 0$, with the parameters from Section 4.1), attack frequency rises from 0.27 to 0.34 as the shared model advances from GPT-4o to Mythos, while defender welfare moves little on net, from −13.79 to −13.22. Scaling both sides equally intensifies attacks without improving defender welfare, a Red Queen's race driven by the pipeline-length asymmetry.

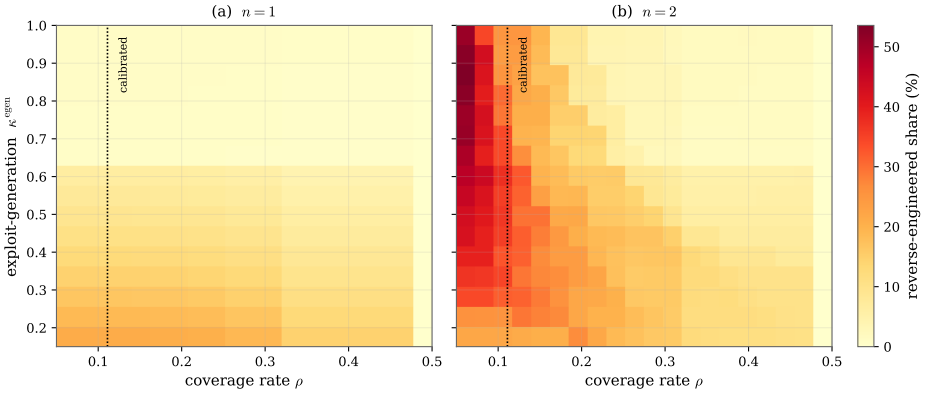

Fig. 4. Patch-diffing share over ρ and κ^{egen}, for (a) $n = 1$ and (b) $n = 2$. Governed by coverage (peaking at low ρ, gone by $\rho \gtrsim 0.45$); its κ^{egen}-dependence flips with the surface, and the peak rises from 21% to 54%. Dotted line: calibrated ρ.

For a single vulnerability, the expected time to traverse a pipeline is $\sum_i 1/\kappa_i$, with one geometric wait for each Bernoulli stage, and the adversary reaches READY sooner than the defender reaches SHIPPED. Increasing shared capability allows both players to speed up their pipeline transitions, but the same speed-up moves the adversary into a damaging state earlier. Because shipping does not immediately eliminate exposure, coverage grows at the capability-independent rate ρ, so an earlier-ready adversary accumulates damage over more rounds before the fleet is patched.

A pre-release instead opens a capability gap (Figure 5b), with the defender operating at $\boldsymbol{\kappa}^{\text{new}}$ while the adversary remains at $\boldsymbol{\kappa}^{\text{prev}}$, so the defender can ship and begin accumulating coverage before the adversary reaches READY. In our calibration, this gap raises defender welfare from -13.22 to -9.47 and lowers attack frequency from 0.34 to 0.26. The welfare gain thus comes from relative capability, not from a higher common level.

Robustness. The LLM-Delphi panel introduces epistemic uncertainty, reflected in the inter-expert standard deviations reported in Table 2. We propagate this uncertainty by Monte Carlo to assess its effect on the optimal release strategy. For each model transition we draw the previous ($\boldsymbol{\kappa}^{\text{prev}}$) and new ($\boldsymbol{\kappa}^{\text{new}}$) capability vectors, sampling each component independently from a Gaussian with its elicited mean and standard deviation, truncated to $[0, 1]$, and re-solve at $n = 1$. A pre-release remains optimal at both $n = 1$ and $n = 2$ throughout the examined cost range (Figure 6b), although the window is longer at $n = 2$, 24 rather than 12 rounds at the operating point, the value we report for the headline. Across all transitions a pre-release is the modal decision, selected in 80% to 95% of draws, most decisively for the largest offensive jump, Opus 4.6 → Mythos, at 95%, with a median window of 12 rounds ($[9, 15]$) (Table 3).

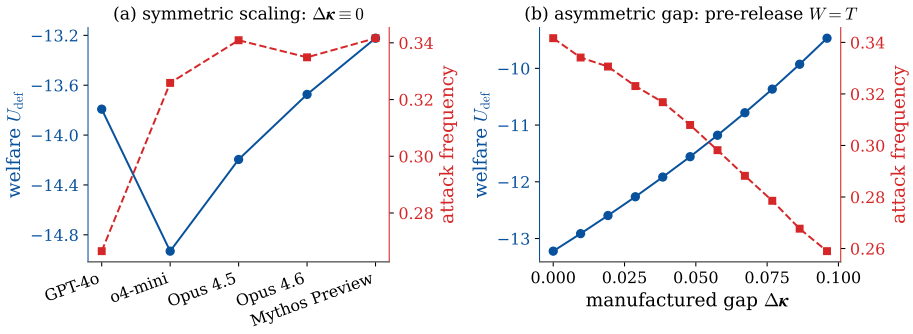

Fig. 5. Welfare turns on the capability *gap*, not the *level*. (a) Symmetric scaling ($\Delta\boldsymbol{\kappa} \equiv 0$): raising the shared level yields no net welfare gain up the ladder, while attacks rise. (b) Asymmetric gap: defender fixed at the Mythos frontier, adversary swept back toward Opus 4.6.

Table 3. Release decision under calibrated uncertainty ($N = 150$ draws per transition, $n = 1$). $P(\mathsf{PRE})$ is the share of draws for which a pre-release is optimal. $W^\star$ is the median pre-release window and Δu_{Def} the mean defender-welfare gain relative to public release, both reported with $[p_{10}, p_{90}]$ intervals. Attacks averted is the mean reduction in attack frequency, relative to public release.

Release transition	$P(\mathsf{PRE})$	$W^\star$ $[p_{10}, p_{90}]$	Δu_{Def} $[p_{10}, p_{90}]$	Attacks averted
GPT-4o → o4-mini	0.873	25 [8, 36]	+4.06 [0.15, 8.43]	41%
o4-mini → Opus 4.5	0.853	19 [9, 25]	+2.94 [0.00, 5.56]	32%
Opus 4.5 → Opus 4.6	0.800	11 [2, 15]	+1.31 [0.00, 2.71]	14%
Opus 4.6 → Mythos Prev	0.953	12 [9, 15]	+2.17 [0.66, 3.66]	23%

The release decision. Discovery κ^{disc} and exploit generation κ^{egen}, the capabilities most influential for the adversary (Figure 3), increase most sharply in the Opus 4.6 → Mythos Preview transition (Table 2). Anthropic ran a pre-release program for this transition. Project Glasswing [2] gave selected partners early access to the then-unreleased Mythos Preview for vulnerability detection. Mozilla reportedly used that access to identify and remediate 271 Firefox vulnerabilities before public release, roughly ten times the number found under Opus 4.6 [20]. This transition is therefore a natural case to examine in detail. We solve it at the calibrated operating point of $n = 2$ vulnerabilities (Section 4.1), where the adversary is fully two-route (Figure 4). Over $\mathcal{A}_{\mathsf{Lab}}$, the lab's Stackelberg solution is a pre-release window of $W^\star = 24$ rounds. Under our calibration, this corresponds to approximately 168 days of exclusive defender access. Relative to public release, it preserves +3.48 welfare and reduces attack frequency from 0.34 to 0.27 (Figure 6).

Our framework provides a structural interpretation of programs such as Glasswing. It identifies the capability gap as the welfare driver, predicts gains

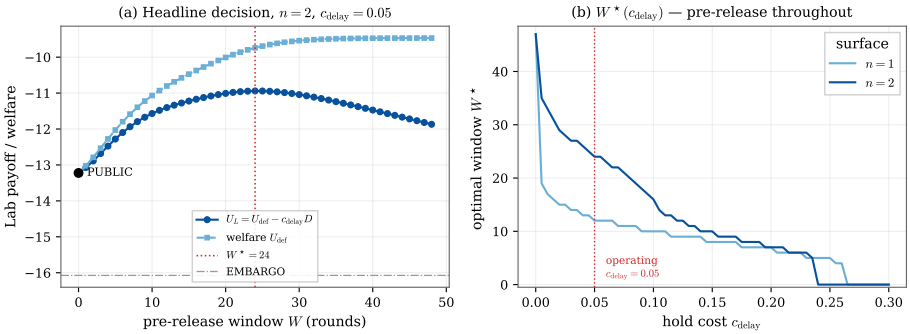

Fig. 6. Calibrated decision for Opus 4.6 → Mythos ($n = 2$, $c_{\text{delay}} = 0.05$). (a) Lab payoff U_{Lab} and welfare u_{Def} over the window W, peaking at the interior $W^\star$=24, above public release and embargo. (b) Optimal window $W^\star$ over the hold cost c_{delay}, a pre-release beneficial for both n=1 and n=2.

for offense-weighted releases, and converts the qualitative case for early access into a model-implied window. The round-to-wall-clock mapping is approximate, so the calibration identifies a welfare-beneficial strategy rather than a precise duration. The strategy is nevertheless robust, with a pre-release optimal in 80% to 95% of draws for every transition in Table 3. With monetary values for per-round damage $d^{(v)}$ and delay cost c_{delay}, which are treated here as a numeraire and a swept parameter, respectively, the model could express both the welfare gain and the optimal window in economic terms.

5 Discussion and Limitations

Our model provides a tractable framework for analyzing how frontier AI release policies shape defender welfare, but this tractability rests on several theoretical concessions and empirical approximations.

Game-theoretic abstractions. To guarantee a unique computable value, the inner game relies on three simplifications: full observability, a zero-sum payoff, and a finite horizon. The zero-sum reward assumes a pure damage-maximizing adversary and so establishes an upper bound on harm, since a real adversary bearing operational costs would attack less. Full observability lets the defender prioritize vulnerabilities by the adversary's progress and time patch shipping to limit exposure to reverse engineering, advantages a real defender would lack; this overstates the defender's strength but preserves the release decision, since protection derives from the capability gap rather than from information control. Finally, a finite horizon bounds the game to a single release cycle: each generation's capabilities are fixed per-round rates, and the policy sets only *which* generation each side can access at each round, so AI advancement enters as the discrete $\boldsymbol{\kappa}^{\text{prev}} \to \boldsymbol{\kappa}^{\text{new}}$ jump.

Calibration. The delay and action costs cannot be grounded in public data, so we sweep them and report the window as a function of cost (Figure 6b) rather

than a single value; the reported windows are therefore conditional on the chosen operating point, unlike calibrated parameters such as the adoption rate ρ. The capability rates $\boldsymbol{\kappa}$ are elicited by the LLM-Delphi procedure of Lorenz and Fritz [24] rather than directly measured, following prior work that elicits risk estimates from expert panels [7]; because the procedure maps benchmark pass rates to per-round rates, it could also accommodate evidence from scaffolded agentic systems, where frontier capability increasingly resides. The elicited values determine how large the protective gap is, not whether it protects; the latter holds across the elicited range (Section 4.2). Finally, the round-to-wall-clock mapping rests on sector-average remediation data and is unvalidated against any specific rollout, so the framework identifies an optimal release strategy, not a precise duration.

Future work. The model identifies the welfare-optimal pre-release window, but the lab weighs its private cost of delay against defender welfare, so the window it chooses can fall short of the social optimum. Aligning the two through mechanism design is the extension this work most directly motivates. Beyond it, the model can be extended toward real-world dynamics: heterogeneous defenders, richer release instruments such as security guardrails and performance throttling, repeated releases, and competition between labs.

6 Conclusion

Responsible disclosure has long given defenders a head start in the vulnerability lifecycle. Frontier AI erodes that advantage by granting both sides the same model, pushing the release decision upstream to the labs that build them. We model that decision as a bilevel three-player Stackelberg game whose leader determines the release strategy, setting both sides' capabilities and parameterizing a downstream zero-sum stochastic game between defender and adversary.

Defender welfare depends on the AI capability *gap* between defender and adversary, not on the absolute capability level. Because a patch must be built and tested while an exploit need only be weaponized, the defender's extra pipeline steps turn equal AI capability gains into a smaller end-to-end speedup than the adversary's, so a timed pre-release of a frontier AI model to the defender creates a protective gap. Calibrated to successive frontier-model transitions via an LLM-Delphi elicitation method, a pre-release is the lab's equilibrium choice in 80–95% of posterior draws.

For models that clear the deployment threshold but carry dual-use capability, the protective lever is sequencing access rather than withholding it entirely. Today's threshold-based safety frameworks do not capture this graduated control. Aligning the lab's private delay cost with the social optimum through mechanism design is the natural next step.

Acknowledgments. This work was partially funded by the European Union under Horizon Europe grants No.101214398 (ELLIOT) and No.101070617 (ELSA). Funded by the European Union. Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union or the European Commission. Neither the European Union nor the European Commission can be held responsible for them. It was also partially supported by the German Federal Ministry of Education and Research (BMBF) under grant AIgenCY (16KIS2012).

References

1. AI Security Institute: Our evaluation of claude mythos preview's cyber capabilities (2026), https://www.aisi.gov.uk/blog/our-evaluation-of-claude-mythos-previews-cyber-capabilities
2. Anthropic: Project Glasswing: Securing critical software for the AI era (Apr 2026)
3. Anthropic: Responsible scaling policy, version 3.3 (2026)
4. Arora, A., Telang, R., Xu, H.: Optimal policy for software vulnerability disclosure. Management Science **54**(4), 642–656 (2008)
5. Axelrod, R., Iliev, R.: Timing of cyber conflict. Proceedings of the National Academy of Sciences **111**(4), 1298–1303 (2014)
6. Bao, T., Shoshitaishvili, Y.: Cyber autonomy in software security: Techniques and tactics. Game Theory and Machine Learning for Cyber Security pp. 204–229 (2021)
7. Barrett, S., Murray, M., Quarks, O., Smith, M., Kryś, J., Campos, S., Boria, A.T., Touzet, C., Hayrapet, S., Heiding, F., et al.: Toward quantitative modeling of cybersecurity risks due to ai misuse. arXiv preprint arXiv:2512.08864 (2025)
8. Brumley, D., Poosankam, P., Song, D., Zheng, J.: Automatic patch-based exploit generation is possible: Techniques and implications (2008)
9. Canann, T.J.: Toward a theory of vulnerability disclosure policy: a hacker's game. In: International Conference on Decision and Game Theory for Security. pp. 118–134. Springer (2019)
10. Charrier, C., Weiner, R.: How low can you go? an analysis of 2023 time-to-exploit trends. Tech. rep., Mandiant (2024)
11. Choi, J.P., Fershtman, C., Gandal, N.: Network security: Vulnerabilities and disclosure policy. The Journal of Industrial Economics **58**(4), 868–894 (2010)
12. Dai, D., Liu, M., Li, A., Cao, J., Wang, Y., Wang, C., Peng, X., Zheng, Z.: Feedbackeval: A benchmark for evaluating large language models in feedback-driven code repair tasks. arXiv preprint arXiv:2504.06939 (2025)
13. Dalkey, N., Helmer, O.: An experimental application of the delphi method to the use of experts. Management science **9**(3), 458–467 (1963)
14. DeepMind, G.: Frontier safety framework, version 3.1 (2026)
15. Edgescan: 2025 vulnerability statistics report. Tech. rep., Edgescan (2025)
16. Epp, S.: Zero day clock (2026), https://zerodayclock.com/
17. Fang, R., Bindu, R., Gupta, A., Kang, D.: Llm agents can autonomously exploit one-day vulnerabilities. arXiv preprint arXiv:2404.08144 (2024)
18. Flake, H.: Structural comparison of executable objects. In: Detection of intrusions and malware & vulnerability assessment, GI SIG SIDAR workshop, DIMVA 2004. pp. 161–173. Gesellschaft für Informatik eV (2004)
19. Halawi, D., Zhang, F., Yueh-Han, C., Steinhardt, J.: Approaching human-level forecasting with language models. Advances in Neural Information Processing Systems **37**, 50426–50468 (2024)
20. Holley, B.: The zero-days are numbered (2026), https://blog.mozilla.org/en/privacy-security/ai-security-zero-day-vulnerabilities/
21. Hutchins, E.M., Cloppert, M.J., Amin, R.M.: Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. Tech. rep., Lockheed Martin Corporation (2011), white paper
22. Jimenez, C.E., Yang, J., Wettig, A., Yao, S., Pei, K., Press, O., Narasimhan, K.R.: SWE-bench: Can language models resolve real-world github issues? In: The Twelfth International Conference on Learning Representations (2024)

23. Kamhoua, C.A., Kiekintveld, C.D., Fang, F., Zhu, Q.: Game theory and machine learning for cyber security. John Wiley & Sons (2021)
24. Lorenz, T., Fritz, M.: Scalable delphi: Large language models for structured risk estimation. arXiv preprint arXiv:2602.08889 (2026)
25. Moore, T., Friedman, A., Procaccia, A.D.: Would a'cyber warrior'protect us: exploring trade-offs between attack and defense of information systems. In: Proceedings of the 2010 New Security Paradigms Workshop. pp. 85–94 (2010)
26. Mündler, N., Müller, M.N., He, J., Vechev, M.: Swt-bench: Testing and validating real-world bug-fixes with code agents. In: Globerson, A., Mackey, L., Belgrave, D., Fan, A., Paquet, U., Tomczak, J., Zhang, C. (eds.) Advances in Neural Information Processing Systems. vol. 37, pp. 81857–81887. Curran Associates, Inc. (2024)
27. Murray, M., Papadatos, H., Quarks, O., Gimenez, P.F., Campos, S.: Mapping ai benchmark data to quantitative risk estimates through expert elicitation. arXiv preprint arXiv:2503.04299 (2025)
28. OpenAI: Preparedness framework, version 2 (2025)
29. Phuong, M., Aitchison, M., Catt, E., Cogan, S., Kaskasoli, A., Krakovna, V., Lindner, D., Rahtz, M., Assael, Y., Hodkinson, S., et al.: Evaluating frontier models for dangerous capabilities. arXiv preprint arXiv:2403.13793 (2024)
30. Rescorla, E.: Is finding security holes a good idea? IEEE Security & Privacy (2005)
31. Schoenegger, P., Tuminauskaite, I., Park, P.S., Bastos, R.V.S., Tetlock, P.E.: Wisdom of the silicon crowd: Llm ensemble prediction capabilities rival human crowd accuracy. Science Advances **10**(45), eadp1528 (2024)
32. Shapley, L.S.: Stochastic games. Proceedings of the national academy of sciences **39**(10), 1095–1100 (1953)
33. Shevlane, T.: Structured access: an emerging paradigm for safe ai deployment. arXiv preprint arXiv:2201.05159 (2022)
34. Solaiman, I.: The gradient of generative ai release: Methods and considerations. In: Proceedings of the 2023 ACM conference on fairness, accountability, and transparency. pp. 111–122 (2023)
35. Solaiman, I., Brundage, M., Clark, J., Askell, A., Herbert-Voss, A., Wu, J., Radford, A., Krueger, G., Kim, J.W., Kreps, S., et al.: Release strategies and the social impacts of language models. arXiv preprint arXiv:1908.09203 (2019)
36. Souppaya, M., Scarfone, K.: Guide to enterprise patch management planning: Preventive maintenance for technology. NIST Special Publication 800-40 Rev. 4, National Institute of Standards and Technology (Apr 2022)
37. Tambe, M.: Security and game theory: algorithms, deployed systems, lessons learned. Cambridge university press (2011)
38. Wang, Z., Schiller, N., Li, H., Narayana, S.S., Nasr, M., Carlini, N., Qi, X., Wallace, E., Bursztein, E., Invernizzi, L., et al.: Exploitgym: Can ai agents turn security vulnerabilities into real attacks? arXiv preprint arXiv:2605.11086 (2026)
39. Wang, Z., Shi, T., He, J., Cai, M., Zhang, J., Song, D.: Cybergym: Evaluating ai agents' real-world cybersecurity capabilities at scale. arXiv preprint arXiv:2506.02548 (2025)
40. Zhang, A., Ji, J., Menders, C., Dulepet, R., Qin, T., Wang, R., Wu, J., Liao, K., Li, J., Hu, J., et al.: Bountybench: Dollar impact of ai agent attackers and defenders on real-world cybersecurity systems. Advances in Neural Information Processing Systems **38** (2026)
41. Zhang, A.K., Perry, N., Dulepet, R., Ji, J., Menders, C., Lin, J., Jones, E., Hussein, G., Liu, S., Jasper, D., et al.: Cybench: A framework for evaluating cybersecurity capabilities and risks of language models. In: International Conference on Learning Representations. vol. 2025, pp. 25094–25243 (2025)