A Security-Oriented Lifecycle Model for Large Language Model Systems

Eleftherios Batzolis[1,2][0000-0002-5411-2716], George Drosatos[2][0000-0002-8130-5775], Vassilis Katsouros[2][0000-0002-4185-2344], and Konstantinos Rantos[1][0000-0003-2453-3904]

[1] Department of Informatics, Democritus University of Thrace, Kavala, Greece
ebatzoli@cs.duth.gr; krantos@cs.duth.gr
[2] Institute for Language and Speech Processing, Athena Research Center, Xanthi, Greece
e.batzolis@athenarc.gr; gdrosato@athenarc.gr; vsk@athenarc.gr

Abstract. Large language models are being integrated into critical infrastructure and enterprise workflows at unprecedented scale,yet the lifecycle frameworks governing their development and operations were designed for operational efficiency rather than security analysis. As a result, security-relevant activities such as data provenance verification, artifact signing, agentic permission control, and decommissioning are often left implicit or assumed to receive due care. Governance frameworks, in turn, organise requirements around risk levels or management processes without clearly linking them to the lifecycle stages where they apply. This paper addresses both deficiencies. We propose a lifecycle model for LLM systems that supports security analysis by structuring it around security-relevant boundaries rather than workflow optimisation. The model comprises 32 stages across four core pipeline layers (Data, Model, Distribution, Application), supported by a 12-stage LLMOps pillar and a 9-category governance pillar. Thirteen stages are introduced here as separate units because they expose distinct security concerns that existing frameworks do not clearly distinguish. A governance mapping synthesising the NIST AI RMF, the EU AI Act, and ISO/IEC 42001 reveals a structural property of the current regulatory landscape: governance evidence concentrates at deployment-facing stages, where systems are visible to regulators, while the most consequential decisions, data selection, alignment strategy, and capability boundaries, are made at development-facing stages, where regulatory visibility is lowest.

Keywords: LLM security · AI lifecycle · AI governance · attack surface · supply chain security · LLMOps

1 Introduction

Large Language Models (LLMs) have evolved from research artifacts into foundational components of critical infrastructures and enterprise workflows [49].

Their development spans multi-stage chains that include data sourcing and curation, alignment via Reinforcement Learning from Human Feedback (RLHF) [4], post-training optimisation, distribution through public model hubs [39], and deployment within orchestrated stacks incorporating Retrieval-Augmented Generation (RAG), autonomous agents, and multi-model architectures [16]. Each stage introduces distinct artifacts, access models, threat actors, and attack vectors, making the LLM lifecycle a substantially broader attack surface than that of conventional software systems.

The security research community has responded by cataloguing threat categories such as data poisoning [6], prompt injection [15], and model extraction and privacy leakage [18], and by developing industry frameworks including the OWASP Top 10 for LLM Applications [31] and MITRE ATLAS [24]. In parallel, governance instruments such as the NIST AI RMF [25] and ISO/IEC 42001 [20] have established organisational controls intended to fortify AI deployments against malicious activity. What remains absent, however, is a lifecycle structure that connects these perspectives: a model that decomposes the LLM lifecycle at the boundaries where security-relevant distinctions arise, thereby enabling systematic analysis of threats, governance provisions, and operational requirements at appropriate granularity.

Existing lifecycle and governance frameworks, surveyed in Section 2, exhibit a consistent pattern: stage boundaries are defined by engineering workflow or by regulatory logic, neither of which aligns cleanly with security-relevant distinctions. The result is a four-fold gap. First, a *granularity gap*: activities associated with different threat profiles are merged under broader categories, hiding important security differences. Second, a *security-specificity gap*: security-relevant activities such as model signing, red teaming, and input validation are not clearly mapped to distinct lifecycle stages. Third, an *architectural completeness gap*: no existing framework combines end-to-end lifecycle coverage with cross-cutting governance and operational support. Fourth, a *governance mapping gap*: governance frameworks are not sufficiently linked to specific lifecycle stages.

We argue that existing LLM lifecycle and governance frameworks are structurally misaligned for security analysis, and that a security-oriented lifecycle model, decomposed along attack-surface boundaries and mapped to governance provisions, makes that misalignment analysable. We address the four gaps with the following contributions:

1. An *LLM lifecycle model that supports security analysis*, comprising 32 stages across four core pipeline layers. Thirteen are identified here as separate lifecycle stages because they expose distinct security concerns not clearly distinguished in existing frameworks. Several of the underlying activities, such as red teaming and synthetic data generation, are already recognised in practice; what is new here is their treatment as separate analytical units for security analysis.
2. A *12-stage LLMOps operational pillar* synthesised from three academic sources [7,36,32] and elevated from a single lifecycle step to a cross-cutting support layer.

3. A *9-category governance pillar* synthesising the NIST AI RMF [25], the EU AI Act [12], and ISO/IEC 42001 [20], that maps governance concerns to lifecycle stages and identifies three governance handoffs at layer boundaries.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 describes the methodology. Section 4 presents the core pipeline. Sections 5 and 6 detail the LLMOps and governance pillars, respectively. Section 7 discusses differentiation and limitations, and Section 8 concludes.

2 Related Work

Existing lifecycle models exhibit a consistent pattern: stage boundaries are typically defined by engineering workflow, which limits their utility for security analysis because activities with different threat profiles are grouped together. MLOps frameworks [22] provide a general pipeline structure but lack granularity for LLM-specific activities such as alignment or prompt engineering. LLMOps definitions [7,36,32] address LLM-specific concerns but merge activities with distinct security profiles, treating alignment and fine-tuning as one stage despite different threat actors and poisoning vectors, and combining packaging and distribution across different trust boundaries. Enterprise blueprints [5] provide operational context but not the stage decomposition needed for per-stage security assessment. These frameworks address the *granularity gap* and *security-specificity gap* only partially.

Security-oriented framings partially address these gaps. Wang et al. [45] provide the broadest empirical coverage (529 CVEs, 13 stages) but merge distinct threat profiles and lack governance mapping. NIST AI 100-2e2025 [42] compresses the upstream pipeline into a single training phase, precluding distinction among annotation, alignment, and distribution stages. ENISA [10] and ETSI EN 304 223 [11] include decommissioning and apply Confidentiality, Integrity, and Availability (CIA) assessment, validating our design choices, but restrict security analysis to high-level policy guidance without per-stage decomposition. MITRE ATLAS [24] provides indispensable threat intelligence but its kill-chain orientation serves threat hunting rather than lifecycle governance mapping. Google's SAIF [14] organises security around component-risk mappings and distinguishes Model Creator from Model Consumer roles; its 2.0 release adds agentic concerns that corroborate the need for our Stage 4.11, but its structure remains risk-oriented rather than lifecycle-oriented and does not map to external governance instruments. Supply-chain studies [44] validate the four-layer structure as a recurring pattern but predate LLM-specific concerns or provide research agendas without per-stage decomposition. Developer-centric studies [27] corroborate the security significance of stages our framework foregrounds but organize findings by developer themes rather than lifecycle stages. These works address the *architectural completeness gap* only partially: ENISA and ETSI achieve end-to-end coverage including decommissioning, but none combines this with cross-cutting operational and governance pillars.

On governance, the three dominant frameworks, NIST AI RMF [25] (functional axis), the EU AI Act [12] (risk-and-role axis), and ISO/IEC 42001 [20] (management-cycle axis), embody structurally distinct logics, none of which maps onto the operational stages of LLM development. Organisations must navigate all three, yet none indicates how its requirements relate to specific lifecycle stages, leaving the *governance mapping gap* unaddressed.

The present work responds to these four gaps by proposing a lifecycle model that combines a core pipeline structured around security-relevant boundaries with cross-cutting LLMOps and governance pillars. Table 1 summarises the comparison across key dimensions, and the differentiation can be articulated concretely along four axes. On *lifecycle granularity*, the proposed model decomposes the LLM lifecycle into four layers and 32 stages, exceeding the resolution of prior treatments that adopt three architectural layers [44], thirteen stages without end-to-end coverage [45], or two phases that collapse the upstream pipeline [42]. On *security-specific decomposition*, the model identifies thirteen stages as separate lifecycle units on the basis of distinct threat actors, exposed artifacts, and access models; prior frameworks treat these activities either implicitly within broader stages or as risk categories rather than lifecycle units, with the partial exception of ENISA [10] and ETSI [11], which include decommissioning but stop short of per-stage decomposition. On *end-to-end coverage*, the model spans data sourcing through decommissioning, matching ENISA and ETSI in scope while exceeding them in granularity; most other frameworks, including SAIF [14] and MITRE ATLAS [24], omit decommissioning entirely. On *governance linkage*, the model provides an explicit provision-to-stage mapping synthesising the NIST AI RMF, the EU AI Act, and ISO/IEC 42001, which no prior framework in Table 1 offers; ENISA and ETSI achieve partial governance linkage but do not map provisions to stages at the resolution required for per-stage accountability analysis.

3 Methodology

The proposed framework is analytically derived rather than empirically measured: it is constructed through structured synthesis of prior lifecycle, security, and governance literature, and provides a scaffold against which empirical security data can subsequently be organised rather than reporting new empirical findings.

The lifecycle model was constructed through analytical synthesis of three source categories: (i) existing lifecycle frameworks (Section 2), which provided the baseline stage inventory; (ii) the supply-chain and security literature[44,13,45] [6,15,18], which documented threats at stages existing frameworks omit; and (iii) governance instruments [25,26,12,20], whose requirements imply lifecycle stages that operational frameworks do not make explicit.

The organising principle is *attack surface exposure*, characterised along three dimensions: the threat actors with access to a stage, the artifacts exposed at that stage, and the access model governing how those artifacts are reached. Stage boundaries are placed where one or more of these dimensions change in

Table 1. Comparison of LLM/AI Lifecycle and Security Frameworks Against the Four Identified Gaps

Source	Lifecycle Granularity	Security-Specific Stages	Evidence Base	Gov. Linkage	End-to-End Coverage
Wang et al. [44]	3 layers (infra, model, app)	No	Theoretical	None	No (no decommissioning stage)
Steidl et al. [40]	4 stages (general AI)	No	Mixed	None	No (no decommissioning stage)
Wang et al. [45]	13 stages, 3 layers	Partial	Empirical	None	No (no decommissioning stage)
NIST AI 100-2e2025 [42]	2 phases	No	Theoretical	None	No (no decommissioning stage)
Patel et al. [33]	3 attack families	No	Mixed	None	No
Huang et al. [19]	Stakeholder view	No	Mixed	None	No
Nguyen et al. [27]	Developer themes	No	Empirical	None	No
Jiang et al. [21]	Issue categories	No	Empirical	None	No
ENISA [10]	Concept to decom.	Partial	Theoretical	Partial	Yes
ETSI EN 304 223 [11]	5 phases, 13 princ.	Partial	Theoretical	Partial	Yes
MITRE ATLAS [24]	Kill chain tactics	Partial	Mixed	None	No
Google SAIF [14]	Risk-oriented (15 risks)	Partial	Mixed	None	No (no decommissioning stage)
This work	**4 layers, 32 stages**	**Yes (13 stages)**	**Mixed**	**Explicit**	**Yes**

Columns map to the four gaps from Section 1: Lifecycle Granularity, Security-Specific Stages, End-to-End Coverage, and Governance Linkage. Evidence Base indicates each work's empirical grounding.

a way that materially alters the security concerns applicable to the activity in question. We distinguish four threat actor classes along this principle. *External adversaries* (e.g., data poisoners, supply-chain attackers) operate without legitimate access, acting through indirect channels such as publicly scraped data or compromised upstream dependencies. *Insiders* (e.g., compromised annotators, preference evaluators) hold privileged access to internal development artifacts and corrupt training or alignment signals at their point of creation. *Supply-chain intermediaries*, further divided into infrastructure operators (e.g., model hub platforms) and dependency maintainers (e.g., library authors), control artifacts in transit between development and deployment. *End users* (e.g., prompt injectors, agent manipulators) hold legitimate credentials and operate within sanctioned interfaces but may abuse those interfaces in ways that external adversaries cannot, because end users interact with the system after deployment-time controls have already been applied; the mitigations applicable to this class (input validation, output filtering, permission control) are correspondingly distinct from those applicable to external adversaries (perimeter and supply-chain controls). The *access model*, namely the type of artifact exposed, the level of privilege available, and the authentication boundary in place, is the third dimension of attack surface exposure and the dimension along which threat actor classes are distinguished in practice. Lifecycle operations on data, models, or deployed systems are grouped into the same stage when they share similar threat actors, exposed artifacts, and access patterns; they are separated when these charac-

teristics differ enough to warrant independent security analysis. This principle operates as a guiding heuristic rather than a formal decision procedure.

As a worked example, the separation of Supervised Fine-Tuning (SFT, Stage 2.3) from Alignment (Stage 2.4), here understood as preference-based fine-tuning that shapes model behaviour toward human preferences and safety constraints [4], follows directly from this principle. SFT poisoning is mounted by data curators against instruction-response pairs through instruction injection, whereas alignment poisoning is mounted by preference evaluators and reward-model trainers against preference rankings and reward models, through label corruption [23] and reward-model backdoors [43]. The three dimensions of attack surface exposure thus differ across the two activities, warranting separate treatment. The same reasoning, applied across the lifecycle, separates other commonly merged pairs (data sourcing from annotation, packaging from signing) and underlies the introduction of thirteen stages whose security concerns are not represented as separate units in existing lifecycle models.

The governance pillar synthesizes provisions from the NIST AI RMF [25] (including AI 600-1 [26]), the EU AI Act [12], and ISO/IEC 42001 [20], selected because they represent three complementary approaches: voluntary guidance, prescriptive regulation, and certifiable standardization. Provisions were inventoried, assigned to the lifecycle stage(s) where they are most relevant, and aggregated into nine categories (defined in Section 6.1). Governance handoffs, that is, transfers of responsibility and accountability, were identified at points where responsibility shifts, where specific evidence must be passed between roles, or where decisions affect later stages of the lifecycle. The structural asymmetry was identified by classifying each provision as either evidence-producing (e.g., documentation, logging, reporting) or decision-shaping (e.g., data selection, alignment strategy, capability boundaries). This classification revealed that evidence-producing provisions concentrate at Layers 3–4, while decision-shaping provisions concentrate at Layers 1–2.

The LLMOps pillar was derived through three-phase synthesis of three primary sources [7,36,32]: consensus identification (six stages appearing in at least two sources), elevation of implicitly treated concerns to named stages (Guardrails and Content Safety, Security and Compliance Automation), and gap closure (four stages absent from the initial inventory). The twelve stages were cross-referenced against practitioner sources [5]; Table 2 shows the derivation provenance.

4 Core Pipeline: Data, Model, Distribution, Application

The lifecycle model comprises four core layers arranged as a sequential pipeline: Data, Model, Distribution, and Application, comprising 32 stages in total, and is supported by two cross-cutting pillars: the LLMOps pillar (Section 5) and the Governance pillar (Section 6). A continuous feedback loop connects the Application layer back to the Data layer; this loop has security implications, as adversarial feedback collected at Layer 4 can be reintroduced into training data

at Layer 1, propagating issues across iterations. Figure 1 presents the complete model. The subsections below define each layer, focusing on the 13 additional stages; stages that refine existing categories are summarised in Figure 1. Each stage is exposed to different types of threats, and representative examples are provided to illustrate these differences.

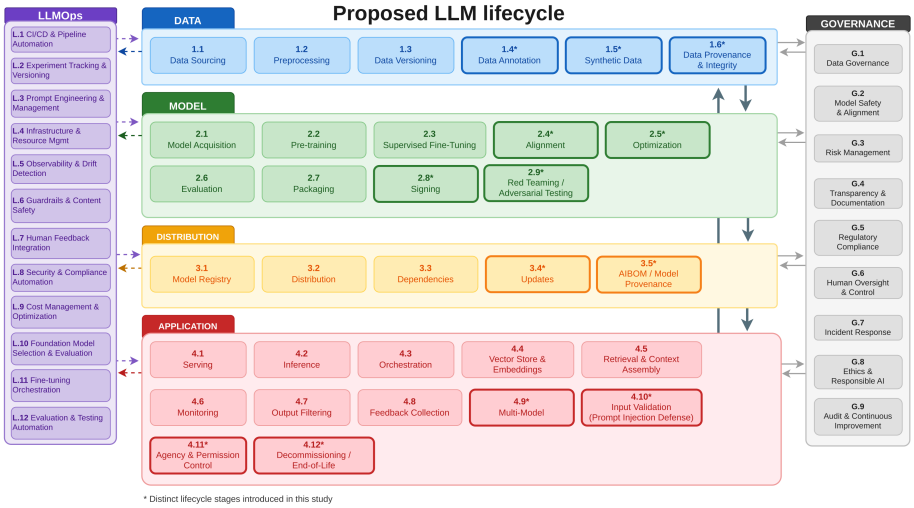

Fig. 1. LLM Lifecycle: four core pipeline layers (Data, Model, Distribution, Application), the LLMOps operational pillar (left), the Governance pillar (right), and the continuous feedback loop.

4.1 Layer 1: Data (Stages 1.1–1.6)

The Data layer produces, curates, and governs the corpora from which an LLM acquires its capabilities; a compromise at any stage of this layer propagates through every subsequent layer [6]. We decompose the Data layer into six stages, two more than typical MLOps frameworks [22] and three more than NIST's monolithic training-data category [42], reflecting differences in security-relevant concerns. Stages 1.1–1.3 (Data Sourcing, Preprocessing, Versioning) refine existing categories. Three additional stages address gaps: *Data Annotation* (1.4) introduces a human-in-the-loop attack surface (label manipulation [1], annotator data exposure, platform injection [34]) materially distinct from automated stages; *Synthetic Data* (1.5) isolates risks unique to model-generated corpora, including model collapse through recursive training [35] and bias amplification through feedback loops; and *Data Provenance and Integrity* (1.6) addresses the metadata infrastructure recording data origins and transformations, which constitutes a bidirectional attack surface: provenance records can be tampered with to disguise poisoned data [2], while overly detailed records may leak sensitive infrastructure information [38].

4.2 Layer 2: Model (Stages 2.1–2.9)

The Model layer spans all activities transforming curated data into a deployable artifact. Its nine-stage decomposition substantially extends prior two- or three-stage treatments [7]. Therefore, we separate acquisition from training (supply-chain vs infrastructure trust), SFT from alignment (different threat actors and poisoning vectors), and introduce dedicated stages for packaging, signing, and adversarial testing, as they involve different security concerns. Stages 2.1–2.3 (Acquisition, Pre-training, SFT), 2.6 (Evaluation), and 2.7 (Packaging) refine existing categories. Four additional stages, introduced here, address gaps: *Alignment* (2.4) is important for model safety, as preference data and reward models can be poisoned to embed backdoors [47], and issues introduced during alignment can weaken safety mechanisms used in later stages. *Optimization* (2.5) addresses post-training transformations (quantization, pruning, merging) where models may appear benign at full precision but behave differently when quantized [9], and compromised models propagate backdoors through weight averaging [48]. *Model Signing* (2.8) addresses cryptographic attestation of artifacts, architecturally distinct from both packaging and distribution, where signing key exposure [37] and the absence of widely adopted standards leave many models distributed without integrity verification. *Red Teaming* (2.9) is separated from general evaluation because it measures robustness under attack rather than capability; red-team findings can also be misused [29], and models can be crafted to evade safety evaluations while remaining harmful in production [49]. Together, these stages make the Model layer the locus of the most consequential and least reversible security decisions: compromises introduced here, whether through poisoned alignment data, backdoored reward models, or absent integrity attestation, propagate through every downstream layer and are seldom remediable by deployment-time controls alone.

4.3 Layer 3: Distribution (Stages 3.1–3.5)

The Distribution layer covers all activities between production of a signed artifact and its instantiation in a running application, which is a major supply-chain attack surface [13]. Stages 3.1–3.3 (Registry, Distribution, Dependencies) refine existing categories. Two additional stages, introduced here, address gaps: *Updates* (3.4) separates post-deployment refresh mechanisms from initial distribution because the trust model differs: updates propagate automatically, often without manual review, and malicious payloads can be delivered through established channels. *AI Bill of Materials (AIBOM) and Model Provenance* (3.5) addresses generation and verification of AIBOMs [30], where AIBOM records may be forged [41] while overly detailed AIBOMs may disclose proprietary details enabling targeted attacks [38].

4.4 Layer 4: Application (Stages 4.1–4.12)

The Application layer spans model instantiation through runtime operation, monitoring, and eventual decommissioning: it is the largest layer reflecting the di-

verse architectural patterns characterising modern LLM deployments. Stages 4.1–4.8 (Serving, Inference, Orchestration, Vector Store, Retrieval, Monitoring, Output Filtering, Feedback Collection) are present in varying degrees across existing frameworks. Serving (4.1) and Inference (4.2), while sharing end users as a threat actor class, differ in artifact type (infrastructure configurations versus inference requests and outputs) and attack vector space (resource exhaustion and configuration tampering versus prompt injection and output manipulation), justifying their separation. Four additional stages, introduced here, address gaps: *Multi-Model* (4.9) captures interaction effects that are not present in single-model deployments, including cross-tenant information channels through shared KV-cache entries that allow one tenant's context to leak into another's generation [46] and adversarial prompt contagion across multi-agent systems [8]. *Input Validation* (4.10) separates defensive infrastructure, such as prompt guards and classification models, from inference, as encoding manipulation and tokenizer-level tricks bypass detection [50] and probing attacks reverse-engineer filter rules. *Agency and Permission Control* (4.11) governs what autonomous agents are permitted to do [29], encompassing tool-use authorization protocols such as the Model Context Protocol (MCP) and Agent-to-Agent (A2A) that define the interface through which agents invoke external capabilities: agents with overly broad permissions can be manipulated into unauthorized actions [28], privilege escalation exploits inter-agent trust [17], and individually permitted tool invocations can be chained to achieve unauthorized outcomes. *Decommissioning* (4.12) addresses model retirement, which is treated explicitly only by a small number of the frameworks reviewed in Table 1: deprecated weights may contain extractable data, orphaned API keys may remain valid, and namespaces of retired models can be claimed by adversaries.

5 The LLMOps Pillar

LLMOps should not be treated as a single step within the Model layer. Prior work [7,36,32] characterises LLMOps as an end-to-end orchestration concern, and the compromise of LLMOps infrastructure carries cascading consequences that exceed those of any single pipeline stage, since it can simultaneously disable enforcement mechanisms and suppress detection. We therefore elevate LLMOps to a cross-cutting pillar comprising twelve sub-stages[3] derived through the three-phase process described in Section 3. We define a sub-stage as *consensus* when it appears as a named function in at least two of the three primary sources, as *elevated* when it is treated implicitly in one or more sources but never named as a distinct operational function, and as *gap-closure* when it is absent from the inventory of all three primary sources. Under this criterion, seven sub-stages are consensus, two are elevated, and three are gap-closure. Together, these sub-stages capture the main operational functions required to manage LLM systems across the lifecycle, and make explicit several aspects that are treated only implicitly in existing definitions. Table 2 summarises how each sub-stage relates to

[3] We use *stage* for core pipeline units and *sub-stage* for LLMOps units throughout.

existing definitions of LLMOps and indicates the synthesis phase from which it was derived.

Table 2. LLMOps Sub-stage Derivation: Source Coverage and Phase

Sub-stage	D[a]	S[b]	P[c]	Phase
L.1 CI/CD & Pipeline Automation	✓	✓	✓	Consensus
L.2 Experiment Tracking & Versioning	✓	✓		Consensus
L.3 Prompt Engineering & Management	✓	✓	✓	Consensus
L.4 Infrastructure & Resource Mgmt	✓		✓	Consensus
L.5 Observability & Drift Detection	✓	✓	✓	Consensus
L.6 Guardrails & Content Safety		∼	∼	Elevation
L.7 Human Feedback Integration	✓	✓		Consensus
L.8 Security & Compliance Automation	∼			Elevation
L.9 Cost Management & Optimization	✓	✓	✓	Consensus
L.10 Foundation Model Selection & Eval.		✓		Gap closure
L.11 Fine-tuning Orchestration		✓		Gap closure
L.12 Evaluation & Testing Automation		✓		Gap closure

[a]Diaz-De-Arcaya [7]; [b]Sinha [36]; [c]Pahune [32]; ✓ = named function; ∼ = implicit treatment.

Every named function in Sinha et al. [36] is represented among the twelve sub-stages, with two exceptions, Infrastructure and Resource Management and Security and Compliance Automation, that are present in other primary sources but not in Sinha, which together suggests that the principal operational functions of LLMOps are accounted for. The relationship between the LLMOps pillar and the core pipeline is bidirectional: in the forward direction, LLMOps orchestrates stage progression, provisions resources, enforces safety constraints, and applies validation checks; in the reverse direction, the pipeline generates the signals, dataset metadata, training metrics, user interactions, and inference costs, on which LLMOps operates. This bidirectional relationship has direct security implications, since controls are applied through LLMOps while security events originate from the pipeline, with the consequence that a single compromise of LLMOps infrastructure can simultaneously degrade enforcement and suppress detection. Consider, for instance, an attacker who compromises CI/CD and Pipeline Automation (L.1): such an attacker may introduce a poisoned model into the deployment workflow, tamper with experiment-tracking and versioning records (L.2) so that the poisoned model appears to pass evaluation, disable the guardrail configurations (L.6) that would otherwise catch anomalous outputs, and suppress the observability alerts (L.5) that would notify defenders. The resulting cascade illustrates how a single entry point can propagate across four LLMOps sub-stages, a pattern that motivates treating LLMOps as a distinct security-analytical layer rather than as a single operational concern.

6 The Governance Pillar

The governance pillar synthesises the NIST AI RMF [25], the EU AI Act [12], and ISO/IEC 42001 [20] into a unified, lifecycle-mapped governance structure. Figure 2 presents the mapping of the governance stack to the LLM lifecycle layers.

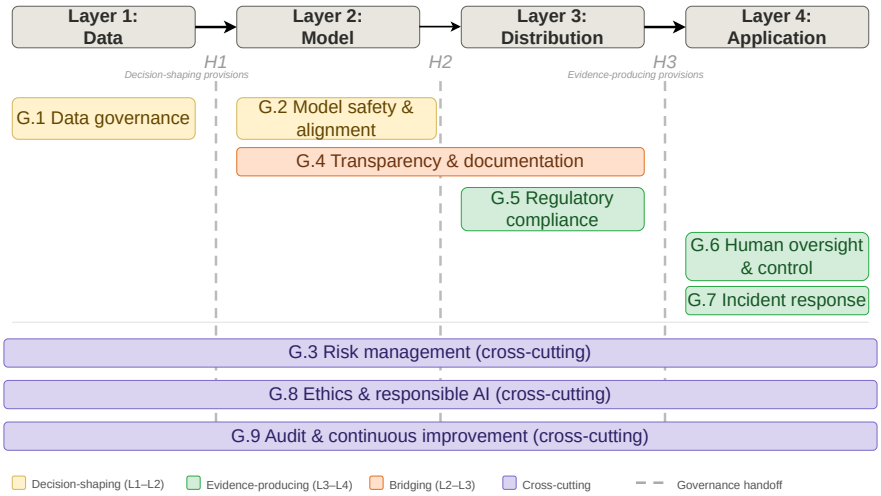

Fig. 2. Governance stack mapped to the LLM lifecycle: nine categories assigned to their primary lifecycle layers.

6.1 Design Rationale and Nine Categories

The three frameworks embody structurally distinct logics: functional (NIST), risk-and-role (EU AI Act), and management-cycle (ISO/IEC 42001), none of which maps directly onto development stages. The governance pillar reframes provisions along a lifecycle axis by assigning each provision to the lifecycle stage at which the activities it regulates actually take place. The EU AI Act, in particular, imposes obligations on providers of *General-Purpose AI (GPAI)* models, that is, foundation models offered for downstream integration; these obligations (Arts. 53–55) are treated here as a distinct sub-class of provisions within the existing categories rather than as a separate governance framework. The nine categories, ordered from data-proximal to cross-cutting, are: Data Governance (G.1), primarily Layer 1; Model Safety and Alignment (G.2), primarily Layer 2; Risk Management (G.3), cross-cutting; Transparency and Documentation (G.4), concentrated at the Layer 2/3 boundary; Regulatory Compliance (G.5), primarily Layer 3, carrying the heaviest enforcement weight (EU AI Act fines up to €15 million or 3% of global turnover); Human Oversight and Control (G.6), primarily Layer 4; Incident Response (G.7), primarily Layer 4; Ethics and Responsible AI (0.8), cross-cutting; and Audit and Continuous Improvement (G.9),

cross-cutting. Table 3 presents a representative excerpt of the provision-to-stage mapping.

Table 3. Governance Provision-to-Stage Mapping

Category	Layer	Key Provisions
G.1 Data Gov.	L1	EU AI Act Art. 10: data governance and quality; GPAI Art. 53(1)(d): training data summary; N: MAP 1, MAP 4: data sources and third-party risks; ISO A.4: data resource controls
G.2 Model Safety	L2	EU AI Act Art. 9: risk management; GPAI Art. 55: adversarial testing; N: MEASURE 1–2: evaluation and risk assessment; GenAI 600-1: hallucination and bias; ISO Cl. 6.1: risk and impact assessment
G.4 Transp. & Doc.	L2/L3	EU AI Act Art. 11: technical documentation; GPAI Art. 53: transparency report; N: GOVERN 4: transparency culture; ISO Cl. 7.5: model cards and AIMS records
G.5 Reg. Compl.	L3	EU AI Act Art. 43: conformity assessment; Arts. 16–17: provider QMS; N: MANAGE 1, GOVERN 1.1: risk treatment and legal requirements; ISO Cl. 8.1: release controls; A.10.3: supplier oversight
G.6 Human Oversight	L4	EU AI Act Art. 14: human oversight; Art. 50: AI disclosure; Art. 27: FRIA; N: GOVERN 5: stakeholder engagement; ISO A.9.3: oversight controls
G.7 Incident Resp.	L4	EU AI Act Art. 12: logging; Arts. 72–73: post-market monitoring and incident reporting; N: MANAGE 4, GOVERN 4.2: residual risk and incident protocols; ISO Cl. 9.1, 10: monitoring and corrective actions; A.8: information for interested parties
G.3 Risk Mgmt	Cross	N: GOVERN (all 6): culture, policy, accountability; EU AI Act: AI Office and MSA oversight; ISO: PDCA cycle; Cl. 5: leadership; Cl. 9: audits
G.8 Ethics	Cross	N: GOVERN 4, MAP 5: ethics and societal impact; EU AI Act Art. 10(2)(g), 10(3): bias and representativeness; ISO A.6: AI system lifecycle controls
G.9 Audit & CI	Cross	N: GOVERN 1, MEASURE: accountability and re-assessment; EU AI Act Arts. 16–17, 72: QMS and post-market monitoring; ISO Cl. 5, 9, 10: leadership, audits, improvement

N = NIST AI RMF; ISO = ISO/IEC 42001; GPAI = General-Purpose AI obligations under the EU AI Act (Arts. 53–55); L1–L4 = Data, Model, Distribution, Application; Cross = cross-cutting. Category order matches Figure 2.

Three features of this distribution are worth noting. The first is that the categories most directly shaping the system, namely Data Governance and Model Safety and Alignment, fall at the upstream end of the lifecycle, where decisions about training corpora, alignment strategy, and capability boundaries are made and where the consequences of those decisions become difficult to reverse. The second is that the categories carrying the heaviest external accountability, namely Transparency and Documentation and Regulatory Compliance, cluster at the boundary between model development and distribution, where evidence produced inside the organisation is handed over to external regulators. The third is that Risk Management, Ethics and Responsible AI, and Audit and Continuous Improvement apply throughout the lifecycle rather than attaching to any one layer; they operate as cross-cutting concerns that interact with every other category.

6.2 Governance Handoffs

Three formal governance handoffs were identified at layer boundaries, with the identification criteria stated in Section 3. The *first handoff* (Layer 1→2) trans-

fers data governance responsibility from data engineering to model development. The data team must produce provenance records, bias assessments, consent documentation, and training data summaries (EU AI Act Art. 10, Art. 53(1)(d)); the model development team consumes these as the evidentiary foundation for model governance. No external enforcement mechanism verifies completeness before consumption, with the consequence that if these artifacts are incomplete, model governance begins on uncertain foundations that downstream testing cannot fully remediate.

The *second handoff* (Layer 2→3) is the most consequential of the three, since at this boundary the provider formally releases the model and accountability transitions to the external regulatory chain. The model development team must transmit conformity declarations (EU AI Act Art. 47), technical documentation (Annex IV), model cards, and, where applicable, GPAI transparency reports (Art. 53) to the distribution function. The EU AI Act's role distinctions become legally operative at this boundary, and releasing without complete documentation propagates irrecoverably into the regulatory chain.

The *third handoff* (Layer 3→4) transfers primary accountability from provider to deployer. The provider retains responsibility for post-market monitoring and incident cooperation (EU AI Act Arts. 72–73), while the deployer assumes accountability for human oversight (Art. 14), logging (Art. 12), and user disclosure (Art. 50). The deployer typically lacks visibility into upstream development decisions, creating an information asymmetry between the entity bearing deployment-time accountability and the entity that made the foundational choices on which that accountability rests.

6.3 Structural Asymmetry

The governance mapping surfaces a notable structural property of the current regulatory landscape: governance obligations and evidence-production requirements concentrate at the deployment-facing end of the lifecycle, while the decisions that most fundamentally determine system security are made at the development-facing end, where regulatory visibility is lowest.

The asymmetry operates along two dimensions. At Layers 1–2, provisions are predominantly decision-shaping, governing what data to collect, what alignment strategy to pursue, and what capability boundaries to impose. These choices are difficult or impossible to reverse once training completes. At Layers 3–4, provisions shift to evidence-producing, requiring documentation, conformity processes, logging, incident reporting, and oversight implementation. Quantitatively, the asymmetry is near-monotonic: of the thirty-three governance provisions mapped to lifecycle-bound categories in Table 3, ten of the eleven provisions assigned to Layers 1–2 are decision-shaping, whereas all twenty-two provisions assigned to Layers 2/3 through 4 are evidence-producing.

At the same time, the EU AI Act places its heaviest enforcement at Layers 3–4, with fines up to €15 million or 3% of global turnover, while the most consequential decisions are made at Layers 1–2, where enforcement is absent. The practical consequence is that an organisation can satisfy every downstream

requirement while having made unverified foundational decisions upstream. This misalignment has direct security implications, since data poisoning, alignment subversion, and capability-boundary violations originate at Layers 1–2, and by the time systems reach Layers 3–4 these compromises are already embedded in model weights. The GPAI provisions (Arts. 53–55) partially bridge this gap by imposing transparency and adversarial-testing requirements on foundation-model providers, but they do not resolve it: the provisions remain evidence-producing rather than decision-shaping, and they do not subject upstream design choices to external verification.

7 Discussion

The model proposed in this paper defines 32 core lifecycle stages, 12 LLMOps sub-stages, and 9 governance categories. Together, these create 53 points where security and governance questions can be asked. The aim is not to claim that these are the only possible boundaries, but to make important differences visible, for example between packaging and signing, input validation and inference, or alignment and supervised fine-tuning. On *architectural completeness*, the model combines end-to-end coverage from data sourcing through decommissioning with both an operational and a governance pillar as cross-cutting structures, a combination we did not identify among the frameworks surveyed in Table 1. On *security-specific decomposition*, the 13 stages introduced here as distinct lifecycle stages fall into three categories: security-critical activities previously subsumed under broader categories (1.4, 1.5, 2.4, 2.5), lifecycle activities omitted entirely (1.6, 2.8, 2.9, 3.4, 3.5, 4.12), and emerging architectural paradigms (4.9, 4.10, 4.11). The *governance mapping* also points to an important imbalance. Many of the most important security decisions are made early, during data and model development, but much of the formal evidence, documentation, and accountability appears later, around distribution and deployment.

Three incident mappings illustrate this finer decomposition's analytical value. These examples should be read as illustrative mappings, not as empirical validation of the model. First, the 2024 Hugging Face serialization attacks [3] map to Stages 2.7–3.2, a distinction between packaging, signing, and registry that two-phase models collapse into a single category, obscuring where the trust boundary was breached. Second, reward model poisoning attacks [43] target Stage 2.4 (Alignment) specifically; frameworks that merge alignment with SFT cannot distinguish whether the poisoning vector is instruction data or preference labels, yielding different mitigations. Third, agentic exploitation attacks target Stage 4.11 (Agency and Permission Control) through tool-use protocols such as MCP, a stage no framework reviewed in Section 2 treats separately.

The model also has several limitations. The most fundamental is complexity: 53 addressable units provide resolution for per-stage analysis but impose cognitive cost. For this reason the model can be adopted gradually using a three-tier adoption model: Tier 1 adopts the four-layer abstraction and three governance handoffs as an organizational orientation tool; Tier 2 adds the 13 additional

stages as priority extensions to existing frameworks; Tier 3 adopts the full 32-stage decomposition with LLMOps pillar as organizational security maturity increases. The framework is intentionally detailed: organisations that skip stages inherit the residual risk that the skipped stage was designed to address, and the framework makes this risk explicit at the point of omission.

The continuous feedback loop from Layer 4 to Layer 1 is not assigned its own stage but is governed by three distributed components: Stage 4.8 (Feedback Collection), Stage 1.1 (Data Sourcing, which consumes feedback as retraining data), and LLMOps sub-stage L.7 (Human Feedback Integration, which orchestrates the loop); dedicated security analysis of the loop's properties is distributed across these units. Certain stages are LLM-specific (alignment, prompt engineering, agentic orchestration); extending to other foundation model types would require modifications.

The governance mapping is interpretive. Other analysts may assign some provisions to neighbouring stages, especially broad provisions related to risk management or ethics. However, the main pattern is likely to remain: early lifecycle stages shape the system, while later stages produce much of the evidence used for oversight and accountability. The stage separation criterion is conservative by design and may cause under-decomposition where security-relevant distinctions exist but lack sufficient published research. Internal validity is bounded by the interpretive nature of stage derivation; external validity by focus on the current regulatory landscape; construct validity by the analytical rather than empirical stage derivation, which a companion study is designed to validate through systematic mapping of security concerns to the lifecycle structure.

8 Conclusion

This paper has presented a lifecycle model for LLM systems, designed to support security analysis, comprising 32 stages across four core pipeline layers organised by attack surface exposure, introducing 13 stages first identified here as distinct lifecycle stages, absent from all surveyed frameworks; a 12-stage LLMOps pillar elevated to a cross-cutting layer whose bidirectional relationship with the pipeline ensures dedicated security analysis of operational infrastructure; and a 9-category governance pillar synthesising the NIST AI RMF, the EU AI Act, and ISO/IEC 42001 into a lifecycle-mapped governance structure that identifies three governance handoffs and reveals a structural property of the current regulatory landscape, a misalignment between where governance evidence concentrates (Layers 3–4) and where the most consequential governance decisions are made (Layers 1–2). Future work will focus on a literature review to map known threats, attacks, and security controls to each stage of the proposed lifecycle. This would help validate the usefulness of the stage boundaries and support more practical per-stage security analysis. Additional priorities include practitioner validation, multi-modal extension, and upstream governance attestation mechanisms extending conformity-assessment logic to the development-facing stages where the most foundational security compromises originate.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

Acknowledgments. This work received partial funding from the European High-Performance Computing Joint Undertaking (JU) under Grant Agreement No. 101234269 for the Pharos AI Factory project, as well as from the Greek Ministry of Digital Governance and Artificial Intelligence.

References

1. Aryal, K., Gupta, M., Abdelsalam, M.: Analysis of label-flip poisoning attack on machine learning based malware detector. In: IEEE/ACM Int. Conf. Automated Software Engineering (ASE) (2022). https://doi.org/10.1109/BigData55660.2022.10020528
2. Baracaldo, N., Chen, B., Ludwig, H., Safavi, J.A.: Mitigating poisoning attacks on machine learning models: A data provenance based approach. In: Proc. AISec Workshop (2017)
3. Casey, B., Santos, J.C.S., Mirakhorli, M.: A large-scale exploit instrumentation study of ai/ml supply chain attacks in hugging face models (2024)
4. Chaudhari, S., et al.: Rlhf deciphered: A critical analysis of reinforcement learning from human feedback for llms. ACM Computing Surveys **58**(2) (2026). https://doi.org/10.1145/3743127
5. Chernigovskaya, M., Walia, D.S., Neumann, K., Hardt, A., Nahhas, A., Turowski, K.: Towards a standardized business process model for llmops. In: Proc. 27th Int. Conf. Enterprise Information Systems (ICEIS). vol. 1, pp. 856–866. SciTePress (2025)
6. Cinà, A.E., et al.: Wild patterns reloaded: A survey of machine learning security against training data poisoning. ACM Computing Surveys **55**(13s) (2023). https://doi.org/10.1145/3585385
7. Diaz-De-Arcaya, J., López-De-Armentia, J., Miñón, R., Ojanguren, I.L., Torre-Bastida, A.I.: Large language model operations (llmops): Definition, challenges, and lifecycle management. In: 2024 9th Int. Conf. Smart and Sustainable Technologies (SpliTech). pp. 1–6. IEEE (2024). https://doi.org/10.23919/SpliTech61897.2024.10612341
8. Ding, R., Xu, T., Shen, X., Ding, A.A., Fei, Y.: Moecho: Exploiting side-channel attacks to compromise user privacy in mixture-of-experts llms. In: Proc. 2025 ACM SIGSAC Conf. Computer and Communications Security (CCS) (2025)
9. Egashira, K., Vero, M., Staab, R., He, J., Vechev, M.: Exploiting llm quantization. In: Advances in Neural Information Processing Systems (NeurIPS) (2024)
10. ENISA: Multilayer framework for good cybersecurity practices for ai. Tech. rep., ENISA (2023)
11. ETSI: Securing artificial intelligence (sai); baseline cyber security requirements for ai models and systems. Tech. Rep. ETSI EN 304 223 V2.1.1, European Telecommunications Standards Institute (2025)
12. European Parliament and Council of the European Union: Regulation (eu) 2024/1689 of the european parliament and of the council laying down harmonised rules on artificial intelligence (2024)
13. Gokkaya, B., Aniello, L., Halak, B.: Software supply chain: A taxonomy of attacks, mitigations and risk assessment strategies. J. Inf. Secur. Appl. **97** (2025). https://doi.org/10.1016/j.jisa.2025.104324

14. Google: Secure ai framework (saif) (2023)
15. Greshake, K., Abdelnabi, S., Mishra, S., Endres, C., Holz, T., Fritz, M.: Not what you've signed up for: Compromising real-world llm-integrated applications with indirect prompt injection. In: Proc. 16th ACM Workshop on Artificial Intelligence and Security (AISec '23). pp. 79–90 (2023). https://doi.org/10.1145/3605764.3623985
16. He, F., Zhu, T., Ye, D., Liu, B., Zhou, W., Yu, P.S.: The emerged security and privacy of llm agent: A survey with case studies. ACM Computing Surveys **58**(6) (2026). https://doi.org/10.1145/3773080
17. Hou, X., Zhao, Y., Wang, S., Wang, H.: Model context protocol (mcp): Landscape, security threats, and future research directions. ACM Trans. Software Engineering and Methodology (2025). https://doi.org/10.1145/3796519
18. Hu, H., Salcic, Z., Sun, L., Dobbie, G., Yu, P.S., Zhang, X.: Membership inference attacks on machine learning: A survey. ACM Computing Surveys **54**(11s) (2022). https://doi.org/10.1145/3523273
19. Huang, K., Chen, B., Lu, Y., Wu, S., Wang, D., Huang, Y., Jiang, H., Zhou, Z., Cao, J.: Lifting the veil on composition, risks, and mitigations of the large language model supply chain (2024)
20. International Organization for Standardization: Iso/iec 42001:2023 — artificial intelligence management system (2023)
21. Jiang, W., et al.: An empirical study of artifacts and security risks in the pre-trained model supply chain. In: Proc. 2022 ACM Workshop on Software Supply Chain Offensive Research and Ecosystem Defenses (2022). https://doi.org/10.1145/3560835.3564547
22. Kreuzberger, D., Kühl, N., Hirschl, S.: Machine learning operations (mlops): Overview, definition, and architecture. IEEE Access **11**, 31866–31879 (2023). https://doi.org/10.1109/ACCESS.2023.3262138
23. Kusaka, S., Saito, K., Kudo, M., Tanabe, T., Wachi, A., Akimoto, Y.: Cost-minimized label-flipping poisoning attack to llm alignment (2025)
24. MITRE Corporation: Mitre atlas — adversarial threat landscape for artificial-intelligence systems (2025)
25. National Institute of Standards and Technology: Artificial intelligence risk management framework (ai rmf 1.0). Tech. Rep. NIST AI 100-1, NIST (2023). https://doi.org/10.6028/NIST.AI.100-1
26. National Institute of Standards and Technology: Nist ai 600-1: Artificial intelligence risk management framework: Generative ai profile. Tech. Rep. NIST AI 600-1, NIST (2024). https://doi.org/10.6028/NIST.AI.600-1
27. Nguyen, T.A., Le, T.H.M., Babar, M.A.: Securing the ai supply chain: What can we learn from developer-reported security issues and solutions of ai projects? In: Proc. 2026 IEEE/ACM 48th Int. Conf. Software Engineering (ICSE). pp. 1–13. ACM (2026). https://doi.org/10.1145/3744916.3787796
28. Operant AI: Shadow escape: Zero-click mcp exploit (2025)
29. OWASP: Top 10 for agentic applications (2025)
30. OWASP Foundation: Owasp aibom project (2025)
31. OWASP Foundation: Owasp top 10 for large language model applications 2025 (2025)
32. Pahune, S., Akhtar, Z.: Transitioning from mlops to llmops: Navigating the unique challenges of large language models. Information **16**(2), 87 (2025). https://doi.org/10.3390/info16020087

33. Patel, R., Tripathi, H., Stone, J., Golilarz, N.A., Mittal, S., Rahimi, S., Chaudhary, V.: Towards secure mlops: Surveying attacks, mitigation strategies, and research challenges (2025)
34. Saeeda, H., Mohamad, M., Knauss, E., Horkoff, J., Nouri, A.: Re for ai in practice: Managing data annotation requirements for ai autonomous driving systems (2025)
35. Shumailov, I., Shumaylov, Z., Zhao, Y., Papernot, N., Anderson, R., Gal, Y.: Ai models collapse when trained on recursively generated data. Nature **631**(8022) (2024). https://doi.org/10.1038/s41586-024-07566-y
36. Sinha, M., Menon, S., Sagar, R.: Llmops: Definitions, framework and best practices. In: 2024 Int. Conf. Electrical, Computer and Energy Technologies (ICECET). IEEE (2024). https://doi.org/10.1109/ICECET61485.2024.10698359
37. Sood, A.K., Zeadally, S.: Malicious ai models undermine software supply-chain security. Communications of the ACM (2025). https://doi.org/10.1145/3704724
38. Spoczynski, M., Melara, M.S., Szyller, S.: Atlas: A framework for ml lifecycle provenance and transparency (2025). https://doi.org/10.1109/EuroSPW67616.2025.00058
39. Stalnaker, T., et al.: An empirical analysis of machine learning model and dataset documentation, supply chain, and licensing challenges on hugging face. ACM Trans. Software Engineering and Methodology (2025). https://doi.org/10.1145/3776739
40. Steidl, M., Felderer, M., Ramler, R.: The pipeline for the continuous development of artificial intelligence models: Current state of research and practice. Journal of Systems and Software **199**, 111615 (2023). https://doi.org/10.1016/j.jss.2023.111615
41. Vandendriessche, W.: Aibomgen: Generating an ai bill of materials for secure, transparent, and compliant model training (2026)
42. Vassilev, A.: Adversarial machine learning: A taxonomy and terminology of attacks and mitigations. Tech. Rep. NIST AI 100-2e2025, National Institute of Standards and Technology (2025). https://doi.org/10.6028/NIST.AI.100-2e2025
43. Wang, J., Wu, J., Chen, M., Vorobeychik, Y., Xiao, C.: Rlhfpoison: Reward poisoning attack for reinforcement learning with human feedback in large language models. In: Proc. 62nd Annual Meeting Assoc. Computational Linguistics (ACL). pp. 2551–2570 (2024). https://doi.org/10.18653/v1/2024.acl-long.140
44. Wang, S., Zhao, Y., Hou, X., Wang, H.: Large language model supply chain: A research agenda. ACM Trans. Software Engineering and Methodology **1**(1) (2024). https://doi.org/10.1145/3708531
45. Wang, S., Zhao, Y., Liu, Z., Zou, Q., Wang, H.: Sok: Understanding vulnerabilities in the large language model supply chain (2025)
46. Wu, G., et al.: I know what you asked: Prompt leakage via kv-cache sharing in multi-tenant llm serving. In: Network and Distributed System Security Symposium (NDSS) (2025). https://doi.org/10.14722/ndss.2025.241772
47. Wu, J., Wang, J., Xiao, C., Wang, C., Zhang, N., Vorobeychik, Y.: Preference poisoning attacks on reward model learning. In: IEEE Symposium on Security and Privacy (S&P) (2025). https://doi.org/10.1109/SP61157.2025.00094
48. Zhang, J., et al.: Badmerging: Backdoor attacks against model merging. In: Proc. ACM SIGSAC Conf. Computer and Communications Security (CCS) (2024)
49. Zhao, W.X., et al.: A survey of large language models (2025)
50. Zhu, W., Xiang, Z., Niu, W., Guan, L.: Metabreak: Jailbreaking online llm services via special token manipulation (2026)